

NAVAL POSTGRADUATE SCHOOL

Monterey, California



THESIS

DETECTION AND FEATURE EXTRACTION OF MINE- LIKE OBJECTS FROM SEISMIC SONAR SIGNALS

by

Craig Alan Wilgenbusch

March 2001

Thesis Advisor:
Second Reader:

Monique P. Fargues
Roberto Cristi

Approved for public release; distribution is unlimited.

20010613 050

REPORT DOCUMENTATION PAGE			<i>Form Approved OMB No. 0704-0188</i>	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.				
1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE March 2001	3. REPORT TYPE AND DATES COVERED Master's Thesis	
4. TITLE AND SUBTITLE: Title (Mix case letters) Detection and Features Extraction of Mine-Line Objects From Seismic Sonar Signals			5. FUNDING NUMBERS	
6. AUTHOR(S)			8. PERFORMING ORGANIZATION REPORT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000			10. SPONSORING / MONITORING AGENCY REPORT NUMBER	
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A			10. SPONSORING / MONITORING AGENCY REPORT NUMBER	
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.				
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.			12b. DISTRIBUTION CODE	
13. ABSTRACT (maximum 200 words) This thesis investigates detection and classification issues when dealing with seismic signals and represents a first step in the direction of automated detection and classification of mine-like signals obtained using a seismic approach. A computationally cheap detection scheme that utilizes a combination of a simple combination of a short-term energy and zero-crossing detector is implemented and tested on five different classes of targets, resulting in a 100% detection rate for all non-natural targets and 33% detection rate of mine sized rock buried in sand. Three feature extraction methods are evaluated for their possible use in a Gaussian Mixture Model classifier: higher order moments, pole extraction from impulse response modeling using the Steiglitz-McBride iteration, and Radial Basis Function Modeling of data. These methods demonstrate promising results for use in a classifier. However, only a very limited number of data trials per class was available in this work, and the proposed set-up needs to be further validated with additional data.				
14. SUBJECT TERMS Buried Mine Detection, Buried Mine Feature Extraction, Mine Warfare, Seismic Sonar, Mine Classification			15. NUMBER OF PAGES 102	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL	

NSN 7540-01-280-5500

StandardForm298 (Rev. 2-89)

Prescribed by ANSI Std. Z39-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

**DETECTION AND FEATURE EXTRACTION OF MINE-LIKE OBJECTS
FROM SEISMIC SONAR SIGNALS**

Craig A. Wilgenbusch
Lieutenant, United States Navy
B.S. E.E., University of Virginia, 1994

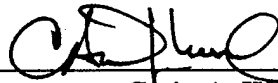
Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

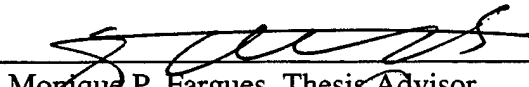
**NAVAL POSTGRADUATE SCHOOL
March 2001**

Author:

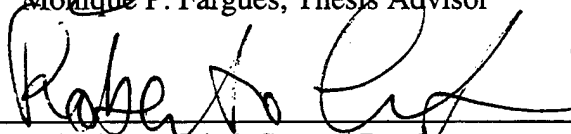


Craig A. Wilgenbusch


Approved by:



Monique P. Fargues, Thesis Advisor



Roberto Cristi, Second Reader



Jeffrey B. Knorr, Chairman
Department of Electrical and Computer Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

This thesis investigates detection and classification issues when dealing with seismic signals and represents a first step in the direction of automated detection and classification of mine-like signals obtained using a seismic approach. A computationally cheap detection scheme that utilizes a combination of a simple combination of a short-term energy and zero-crossing detector is implemented and tested on five different classes of targets, resulting in a 100% detection rate for all non-natural targets and 33% detection rate of mine sized rock buried in sand.

Three feature extraction methods are evaluated for their possible use in a Gaussian Mixture Model classifier: higher order moments, pole extraction from impulse response modeling using the Steiglitz-McBride iteration, and Radial Basis Function Modeling of data. These methods demonstrate promising results for use in a classifier. However, only a very limited number of data trials per class was available in this work, and the proposed set-up needs to be further validated with additional data.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

EXECUTIVE SUMMARY	IX
I. INTRODUCTION.....	1
A. THE MINE PROBLEM	1
1. Naval Mines	1
2. Land Mines and Humanitarian Impacts.....	2
B. MINE DETECTION TECHNOLOGIES	3
1. Current Technologies.....	3
2. New Technologies	3
C. RESEARCH OBJECTIVE AND SUMMARY	4
II. SEISMIC SONAR APPARATUS.....	7
A. PHYSICS OF SEISMIC WAVES	7
1. Rayleigh Wave Properties	7
2. Vector Polarization	10
B. EQUIPMENT AND FIELD ENVIRONMENT	10
1. Self-Contained, Deployable, Seismic Sonar System (SDS ³).....	10
2. Field Environment.....	13
3. Targets and Data Collection.....	15
III. SIGNAL PREPARATION AND DETECTION	19
A. DATA PREPARATION	19
1. Array Processing	19
2. Coherent Subtraction.....	19
3. Vector Polarization Filtering.....	20
4. Results of Coherent Subtraction and Vector Polarization	22
B. DETECTION.....	24
1. Short-term Energy and Zero-crossing Rate Detector.....	25
2. Results/Extracted Target Signals.....	27
C. FAILED METHODS	27
1. Bandwidth Determination	27
2. Spectrograms and Harmonic Correlation	29
3. Kurtosis and Skewness	30
IV. FEATURE EXTRACTION AND CLASSIFICATION	33
A. CLASSIFICATION METHODS	33
1. Feature Extraction	33
2. General Classification and Pattern Recognition Methods	34
B. EVALUATED FEATURE EXTRACTION METHODS	36
1. Higher Order Moments	37
2. Impulse Response Modeling Using STMCB Iteration.....	40
3. Radial Basis Function Modeling of Data.....	42
C. GAUSSIAN MIXTURE MODELS	49
1. Expectation-Maximization Algorithm.....	49
2. Classification Step	51
V. CONCLUSIONS AND RECOMMENDATIONS	53

A.	CONCLUSIONS	53
B.	RECOMMENDATIONS	54
APPENDIX A. MATLAB PROGRAMS (FOR FUNCTIONS CALLED FROM NETLAB TOOLBOX CODE, SEE [REF. 24] FOR CODE)		55
LIST OF REFERENCES		85
INITIAL DISTRIBUTION LIST		87

EXECUTIVE SUMMARY

The United States Navy has historically relied on the projection of naval power ashore which hinges on the ability to perform an expeditious delivery of amphibious forces. In response to the threat of unexploded ordinance and buried land mines, the U.S Navy has called for an improvement in mine detection technology that would allow for a rapid clearing of the surf and beach zone. To meet this challenge, faculty in the Naval Postgraduate Graduate School Physics Department are conducting research to develop a deployable seismic sonar that can rapidly detect and classify mines in the beach zone. Detection and classification issues investigated in this thesis represent a first step in the direction of automated detection and classification of mine-like signals obtained using a seismic approach.

Over the course of several months, experiments were conducted on the U.S. Navy beach in Monterey, CA, to evaluate the effectiveness of different configurations of a seismic sonar apparatus. Signals obtained from those experiments were used to develop an automatic target detection algorithm and evaluate different feature extraction methods. Five different targets were evaluated: a 1000 lb Mk-83 general purpose inert bomb, a 80 lb hollow gas cylinder, a M-19 non-metallic inert anti-tank mine, a standard scuba tank, and a 85 lb rock.

An array of 7 seismic shakers was used to produce Rayleigh waves which were then recorded by an array of 5 seismometers. The signals were processed using array processing, vector polarization filtering, and coherent subtraction techniques. A computationally cheap detection scheme was implemented, using a short-term energy detector in combination with a zero-crossing rate detector to detect and extract the target portion of the signal. The detection scheme was tested on all 5 classes of signals resulting in a 100% detection rate for all non-natural targets and a 33% detection rate on the rock.

Three feature extraction methods were evaluated for their possible use in a Gaussian Mixture Model classifier. Higher order moments, which characterize the distribution in the data, were calculated and plotted, demonstrating and initial clustering

based on two trials per class. Target signals were then modeled as an impulse response of a filter using the Steiglitz-McBride iterative method, evaluating the poles as possible features. The final method used a Gaussian Mixture Model and the Expectation-Maximization (EM) algorithm to model the target signal as a weighted sum of Gaussian shaped pulses, automatically finding the centers, weights, and widths of the pulses. The two features that provided information that distinguished between classes were the weights and the widths. All three methods demonstrate some initial clustering of features that could be used in a Gaussian Mixture Model classifier, but additional trials are needed to validate the proposed methods presented in this thesis.

I. INTRODUCTION

The threat of unexploded ordinance and buried land mines has become an increasingly important problem for militaries and governments worldwide. The threat has existed since the United States Civil war, when the world first saw the effective use of mines used to blockade a harbor, but countermeasure technology has not kept pace. Today, militaries have no practical and expedient method of clearing a beach for an amphibious landing without a significant time, personnel, and equipment investment. Countries faced with the aftermath of war have a formidable task of clearing areas littered with mines and unexploded ordinance that pose a substantial threat to civilian populations. The case for improved mine detection and classification has been repeatedly made in many documents.

A. THE MINE PROBLEM

1. Naval Mines

Of 19 United States ship casualties from 1950-2000, 14 were the direct results of damage sustained from mines [Ref. 1]. The United States Navy has a world class Mine Counter Measure (MCM) force, consisting of ships, aircraft, remote sensors, trained mammals, and explosive ordinance experts, yet it is still not adequate to counter the threat in a safe and as expedient manner as desired.

The importance of expertise in mine warfare was most recently demonstrated in The Gulf War, illustrating the evolving nature of mine warfare and highlighted the requirement for MCM in shallow water. "...From the Sea" and "Forward...From the Sea" by the Chief of Naval Operations (1992 and 1994 respectively) emphasized the importance of warfare in littoral areas [Refs. 2,3]. In "Operational Maneuver from the Sea" (1996), the Commandant of the Marine Corps spelled out the operational concepts of maneuver warfare between sea and land, focusing of the requirement for rapid movement from ship to objective [Ref. 3].

Amphibious landings have always played a major role in every war in the modern era. Defense Guidance and the National Security Act of 1947 require the Marine Corps

to maintain the capability to affect a forcible entry onto a defended shore by means of amphibious assault [Ref. 4]. The projection of naval power ashore, including the effective delivery of U.S. amphibious forces, hinges on the ability to avoid and/or neutralize any possible mine threat. The United States Navy has specified a time frame of six hours in Sea States 0 and 1 (wind speed from 3 to 8 knots and waves up to 1 foot) to conduct the MCM component of an amphibious operation in its near time concept of operations (0 to 10 years). Mid-Far-term concept (10 to 15 years) calls for remote, high speed breaching of a mine field in sea states up to and including Sea State 3 (winds up to 15 knots and waves up to 4 feet high). This will include real time data reporting of detection and mapping of mine fields and obstacles in the beach and surf zones [Ref. 5]. All these requirements and historical examples are compelling reasons to investigate and develop systems that automate the detection and classification process and take human experts out of the loop to facilitate a safe and expeditious landing of an amphibious force.

2. Land Mines and Humanitarian Impacts

The continuing use of cheap and easily attainable landmines continues to be a problem for militaries and civilian populations in the aftermath of a war or conflict. While most governments are aware of the threat, militaries continue to use them, most notably on the Korean Peninsula, where an estimated one million mines divide North from South, Kuwait during the Gulf War, Somalia, and the former Yugoslavia. They provide an easy and effective method of force protection.

Problems arise when a conflict is over and unexploded ordinance remains buried and undocumented. An estimated 100 million unexploded land mines left over from last century's wars and conflicts lie scattered in 64 countries, and it is currently estimated that an additional five million new mines are placed in the ground each year. Even if the laying of land mines were to immediately grind to a halt today, the United Nations estimates that, with conventional methods now being used, it would take over 1,000 years, at a cost of nearly \$33 billion, to safely clear out the world's mine fields [Ref. 6].

B. MINE DETECTION TECHNOLOGIES

Systems exist to sweep and clear mine fields, but only by bulldozing out safe paths and fields. The humanitarian group "People Against Landmines" operates a four-layer system in the countries of Angola, Namibia, and Mozambique. It consists of scrapping off the surface of the earth with converted anti-tank vehicles that can survive mine explosions, fitted with mulchers, then followed up with human and canine detectors [Ref. 7]. This system effectively clears a small safe path, but at the same time has considerable impact on the environment as well as being time inefficient, as a path will be bulldozed regardless if mines are present or not to ensure it is clear.

1. Current Technologies

Today's most advanced in-place technology for the detection of buried mines consist of hand probes, metal detectors, and canine sniffers, all of which put human operators dangerously close to potential mines. In addition, many of today's mines are made from plastic, making them potentially undetectable using metal detectors. Some more technical systems have been developed and field-tested, such as ground penetrating radar (GPR) and infrared imaging [Ref. 7]. GPR is an extremely short-range technology (on the order of one meter), requiring the radar to be placed almost directly over the minefield and infrared imaging can be hampered by dense vegetation cover. The United States Navy also operates a marine mammal program using dolphins, seals, and whales but they cannot operate in the surf and beach zone [Ref. 8].

2. New Technologies

Because of the real danger buried or partially buried mines present to amphibious landing forces, the United States Navy is developing systems that can rapidly detect and classify the threat with low false alarm rates. In the Very Shallow Water (VSW) zone, Surf Zone (SZ), and Shore Zone (SZ) this process presents unique challenges, including low visibility, instability from sea state conditions, and the fact that mines could be buried. Some examples of systems the Navy is developing for this task include bottom-

crawler unmanned underwater vehicles (visual ID system), airborne detection systems (infrared and radar systems), and remote mine-hunting systems deployed from ships (a combination of sonar and visual ID systems). However, none of these possesses the ability to detect all types of mines in all conditions, especially ones that are buried [Ref. 1].

Experiments conducted at the Applied Research Laboratories of the University of Texas at Austin (ARL:UT) and the Physics Department at the Naval Postgraduate School (NPS) have shown that it is possible to detect buried mines using seismic waves generated from a vibrational source. Building on work conducted at ARL:UT [Ref. 9], Fitzpatrick and Hall conducted simultaneous research demonstrating the feasibility of using electromagnetic vibration sources to generate seismic interface, or surface waves. These surface waves then scatter from buried objects and can be used to detect, range, and determine the target strength of such objects [Refs. 10,11]. Sheetz and Guy further developed the concept of the seismic sonar from individual sources and receivers to arrays, providing beam-forming features that maximizes energy along a predetermined axis and the possibility of mechanical or electronic steering of the beam axis. In the process, they developed a more compact and deployable system and evaluated several new electromagnetic sources that proved to be superior in performance [Refs. 12,13].

C. RESEARCH OBJECTIVE AND SUMMARY

Researchers are actively pursuing better methods to detect buried mines and combine them with metal detectors in order to reduce false alarm rates. More innovative methods include GPR, quadrapole resonance, developing mechanical “sniffers” that mimic dogs noses, and a combination seismic and radar apparatus [Ref 7]. It is our goal to add to the mine detection and classification arsenal.

This thesis investigates detection and classification issues when dealing with seismic signals obtained during an on-going project held in the Physics Department at the Naval Postgraduate School. It represents a first step in the direction of automated detection and classification of mine-like signals obtained using a seismic approach. We

investigate a basic and computationally cheap detection scheme and three potential feature extraction schemes that could be used for classification purposes.

THIS PAGE INTENTIONALLY LEFT BLANK

II. SEISMIC SONAR APPARATUS

The operation of the seismic sonar developed by several researchers here at the Naval Postgraduate School (NPS) Physics Department and the Applied Research Laboratories of the University of Texas at Austin (ARL:UT) hinges on several key features of seismic waves. The following section highlights the features of the waves that make them viable for buried land mine detection. A more detailed treatment of the physics behind the seismic sonar and waves is presented in [Refs. 9, 10, 11, 12, 13].

A. PHYSICS OF SEISMIC WAVES

Of the many different types of seismic waves, two main types are of relevance to seismic sonar operation: body waves and surface waves. Only body waves propagate in an infinite, homogeneous, solid elastic, and isotropic medium whereas in an interface region between two different media, both body waves and surface waves exist. Body waves consist of P-waves (primary waves) and S-waves (secondary waves). P-waves are the longitudinal or compression waves and travel faster than S-waves, which are the transverse or shear waves. When body waves (P and S) encounter a solid or an interface of two mediums, they are partially converted into each other. The seismic sonar uses this mixed type surface wave created at the sand/air interface on the beach to detect mines. More specifically, it relies on properties of a specific type of surface wave called a Rayleigh wave, which will be introduced next.

1. Rayleigh Wave Properties

Rayleigh waves occur at the interface between a semi-infinite elastic half-space and a gas, and have both vertical and horizontal particle motion in vertical plane oriented along the direction of propagation. The particles in the wave move in an elliptical trajectory that is retrograde (a rolling motion opposite to the direction of propagation) at the free surface. Below a certain depth, depending on the frequency of the wave, the

particle motion becomes pro-grade elliptical (rolls in the same direction as the propagation), as illustrated in Figure 2.1. This elliptical particle motion exists because

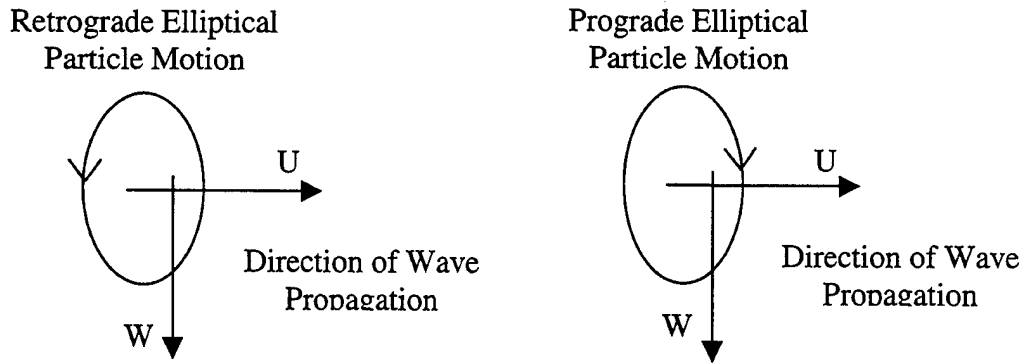


Figure 2.1. Rayleigh wave particle motion. U and W are horizontal and vertical displacement.

of a phase offset in the vertical and horizontal components of the particle motion. In the case of Rayleigh waves, vertical and radial waves are naturally 90 degrees out of phase which allows target detection using a technique called “vector polarization”, as explained in a later section. The depth, h , at which the rotation shifts direction, is approximately $0.1\lambda_R$, where λ_R is the Rayleigh wavelength [Ref. 14].

Figure 2.2 shows the dependence of particle motion on depth, illustrating it is localized to a layer of $2.0\lambda_R$. False returns will not be generated from the substrate or topography of the operating environment and will work just as well on the beach or in the SZ, as illustrated in Figure 2.3. Experiments conducted by Sheetz, Hall and Fitzpatrick as well as others, show that about 67% of the energy produced for a seismic source is found in the Rayleigh waves [Refs. 9,10, 11, 12, 13]. Thus, Rayleigh waves are ideal for seismic sonar applications due their confinement to the surface layer and unique polarization.

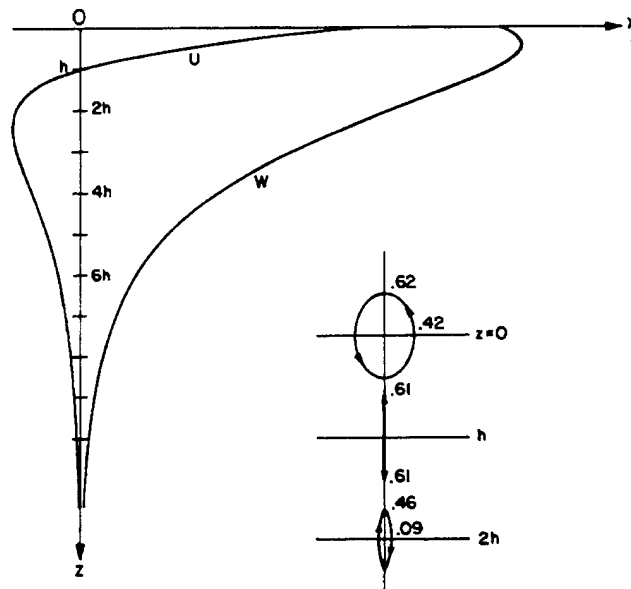


Figure 2.2. The horizontal (U) and vertical (W, down) displacements for Rayleigh waves in a homogenous half-space. U vanishes at depth h. The path of the particles is elliptic retrograde for $z < h$ and elliptic direct (prograde) for $z > h$. From [Ref. 15]

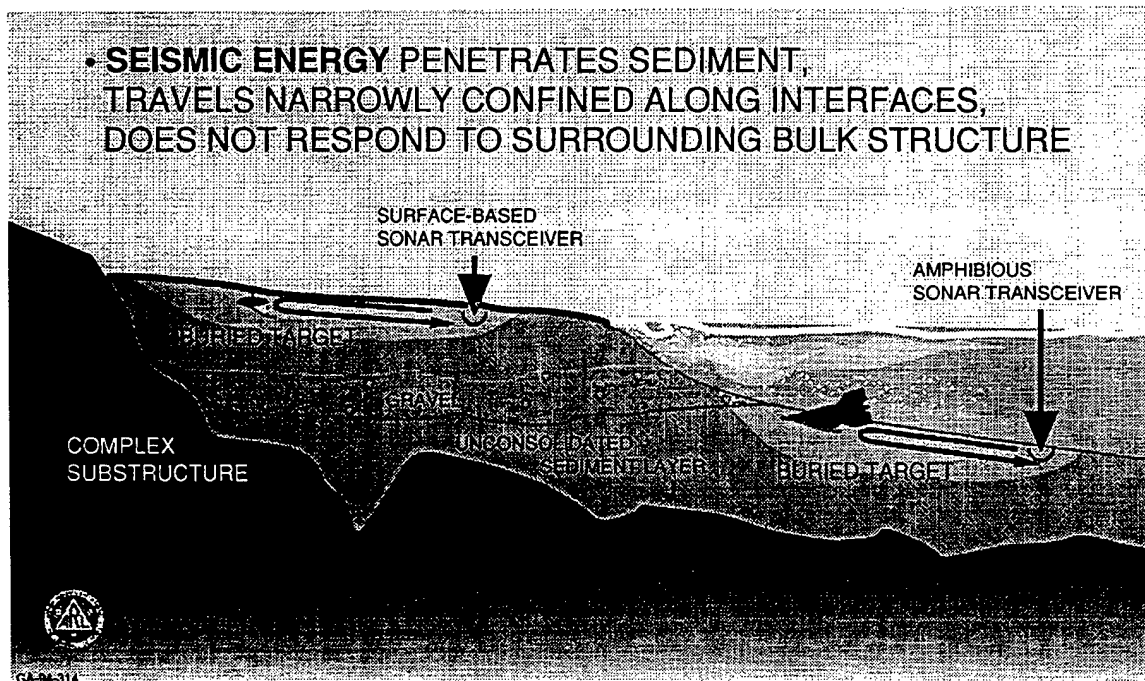
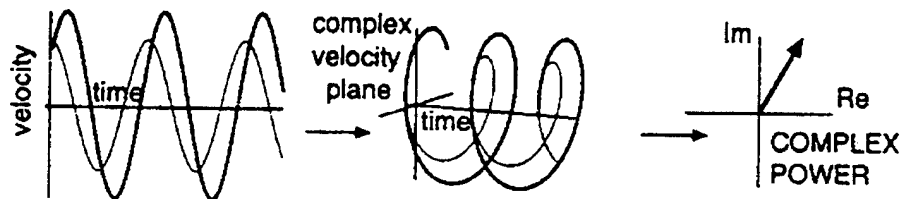


Figure 2.3. Concept for seismic interface wave sonar. From [Ref. 9]

2. Vector Polarization

Vector polarization filtering has successfully been used in the past to extract Rayleigh waves from unwanted body waves (P and S) in a seismic recording because of the 90 degree phase offset exhibited between vertical and radial components, as discussed in the previous section. Note that a phase shift is not present in Body waves, which consist of purely in-phase components. Figure 2.4 shows the difference between the two types of waves and how a phase offset between the two components results in an imaginary component in the complex power for Rayleigh waves and not in P-waves.

Typical Rayleigh Wave:



Typical P-wave noise signal:

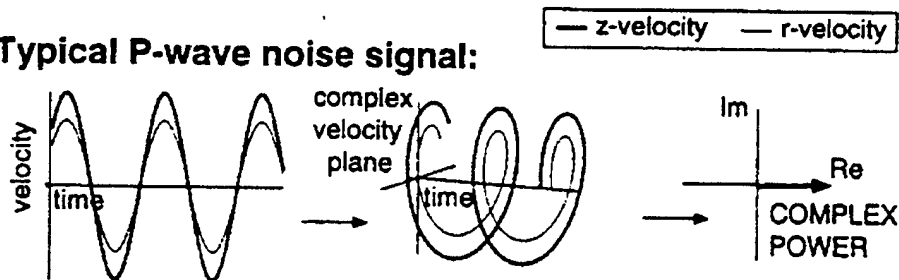


Figure 2.4. Principles of vector vs. scalar wave velocity and complex power relations.
From [Ref. 9]

This unique feature of Rayleigh waves will be used to extract target returns as explained in Chapter III, Section A.3.

B. EQUIPMENT AND FIELD ENVIRONMENT

1. Self-Contained, Deployable, Seismic Sonar System (SDS³)

The current SDS³ research tool is shown in Figure 2.5 fully deployed on the beach. It is self-contained and well suited to access beach environments. Figure 2.6

shows the interior of the system which consisted of a data acquisition system (Signal Processing System's SPS390, a 16 bit, 0-4 kHz analyzer with 8 channels), amplifiers, and

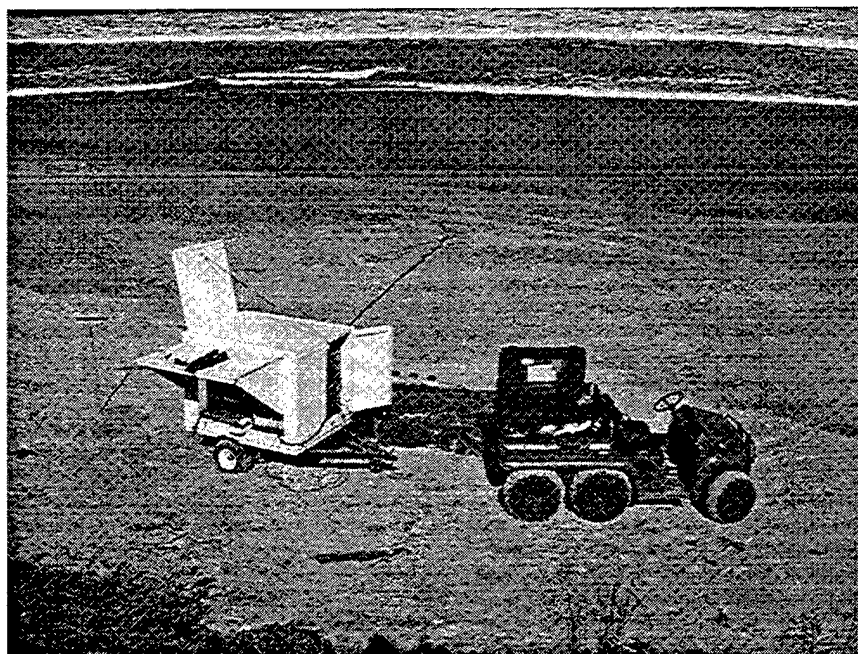


Figure 2.5. Self-contained, Deployable, Seismic Sonar System (SDS³) fully deployed on the Navy Beach, Monterey, California

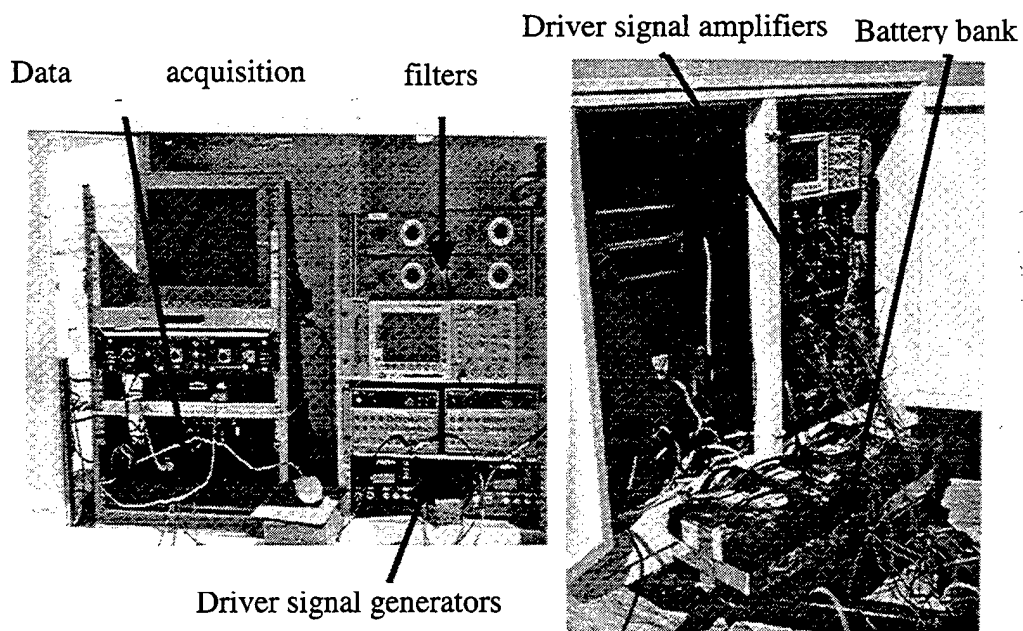


Figure 2.6. Signal processing and amplification equipment inside the trailer

a battery bank. A Honda gasoline generator was used to power the equipment, except for the signal amplifier bank, powered by a bank of car batteries.

The sources developed and tested by Sheetz, Guy, and Muir are shown in Figure 2.7. They are audio base shakers used in car audio systems that are attached back-to-back on a 30.5 cm diameter wooden paddle board and excited by a 100 Hz, high current sinusoidal pulse. These shakers were buried in the sand about two inches below the surface, with approximately 35.6 cm or 14 inches ($1/2$ wavelength estimated for 100Hz at 70 m/s) spacing between shakers. An array of seven shakers is shown in Figure 2.8.

To receive the seismic signals, three-component watertight seismometers, from SENSOR Nederland products (SM-6 4.5 Hz model) were used, as shown in Figure 2.9. Inside each seismometer are three geophones, one each for the x, y, and z axis. A geophone is electro-mechanical device that generates a proportional electrical signal in response to the vibration of a seismic wave. The z axis forms the vertical component of the signal, while the x and y axis together form the radial component. Because of the experimental orientation of the geophones, the radial component is solely comprised of the signal from x-axis geophone, as described in Chapter III, Section A.3. If tilted any more than roughly zero degrees, the response of the geophone suffers severe attenuation, so special attention on the beach was made to ensure they were level. Again, these geophones were extensively tested and configured for use by Sheetz and Guy.

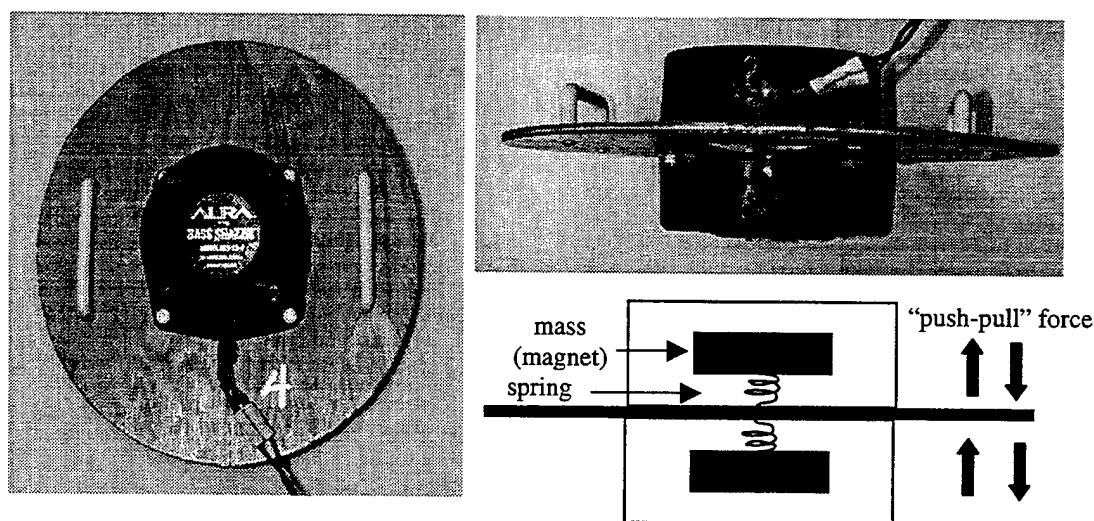


Figure 2.7. Bass shakers used as seismic sources to generate Rayleigh waves.
From [Ref. 12]



Figure 2.8. Seven seismic shaker array

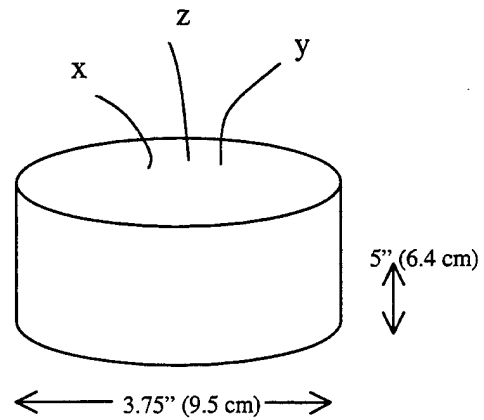


Figure 2.9. Three-component seismometer

2. Field Environment

As previously mentioned, Sheetz and Guy conducted a series of experiments proving beam-forming features that maximizes energy along a predetermined axis can be obtained using an array of seismic sources. Their experimental work shows that indeed it does and also conforms with predicted values, with the results illustrated in Figure 2.10.

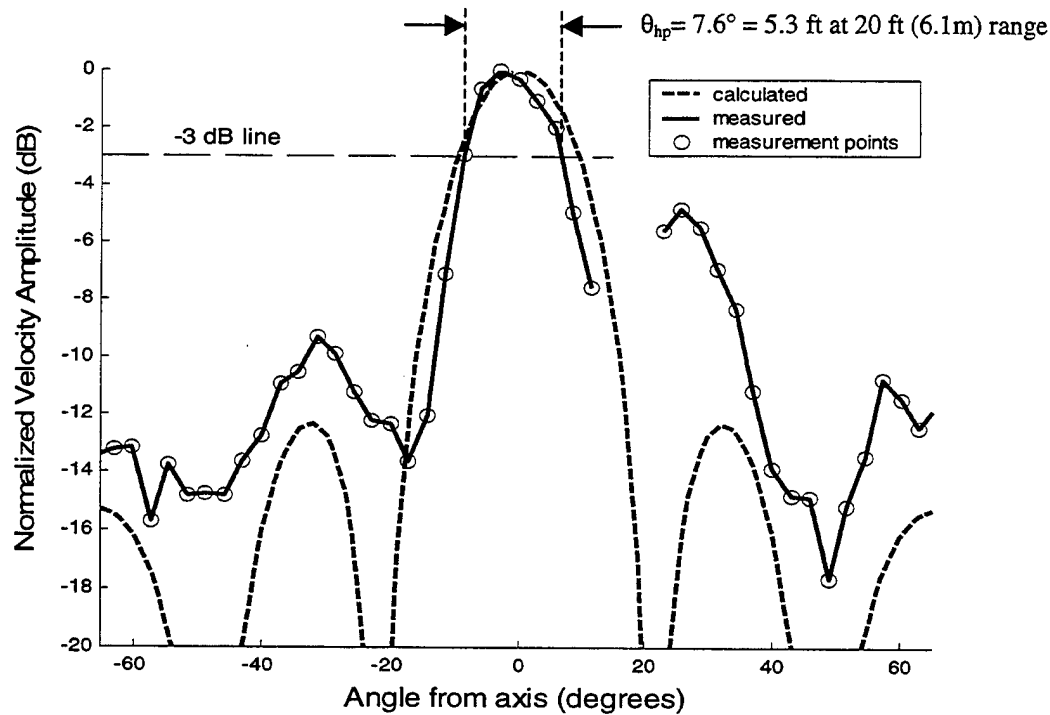


Figure 2.10. Measured (vertical velocity component) and calculated beam pattern at 20 ft (6.1m) range for seven equally spaced, omni-directional elements, driven at uniform amplitude and phase. From [Ref. 12]

The half power angle (-3 dB) of the beam occurs at 6.1 meters (20 ft) with a corresponding beam width of 1.6m (5.3 ft). The reach of energy extends further than previous attempts at applying seismic energy to detect mines, so this avenue appears very promising [Ref. 12].

Based on the previous research of Fitzpatrick, the propagating velocity of Rayleigh waves at 100 Hz was estimated to be 80 m/s. However, further experiments showed that this velocity may vary due to changes in tides, water tables and moisture content of the sand [Ref. 9]. Nevertheless, 80 m/s with an uncertainty value of +/- 10 m/s still remains to be a good estimate.

Sheetz and Guy conducted research on background noise present at the experiment site using a velocity sensitive seismometer. The results they obtained are illustrated in shown in Figure 2.11. The majority of noise is in the 5-20 Hz region, which is well below the source frequency of 100 Hz. As a result, the background noise was

filtered out by passing the recordings through an analog bandpass with a passband of 30-300 Hz before digitization [Ref. 12].

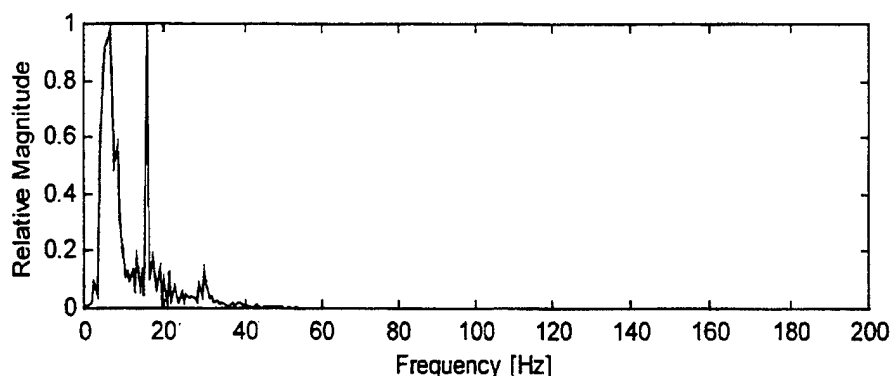


Figure 2.11. Frequency content of background noise at experiment site. From [Ref. 12]

3. Targets and Data Collection

The equipment was set up on the beach in what is referred to as a bi-static configuration, as shown in Figure 2.12. The seismic source array was composed of seven shakers separated by approximately a $\frac{1}{2}$ wavelength, measuring 2.4 m (8 ft) in total length. By placing the five-seismometer array recording array off-axis, the strength of the direct-arrival blast from the source was somewhat mitigated, allowing the geophones to “relax” before the return wave arrival. Seismometer spacing was a $\frac{1}{2}$ wavelength, creating a 1.8 m (6 ft) array. The angle between the orientation of the x-geophone and the axis connecting the seismometer and target is referred to as the “looking angle”, θ . Each seismometer was aligned so that it had a zero degree looking angle so that the x-geophone pointed towards the target.

Figure 2.13 shows some of the targets that were used. The largest, a Mk-63 mine shape, is an inert Mk-83 general purpose 1000 lb (actual weight 1061 lbs) steel bomb that houses mine fusing mechanisms, with dimensions of 2 meters in length and 35 cm in diameter. The second largest target is a hollow gas cylinder, with dimension of 1.32 meters in length and 0.28 meters in diameter, and a weight of 147 lbs. A standard 3000 psi, 80 cubic inch scuba tank was the third smallest target with dimensions of 0.71 meters in length, 0.216 meters in diameter, and a weight of 35 lbs. The M-19 is a square, non-

metallic, blast type anti-tank mine with dimensions 33 cm x 33 cm x 7.5 cm, weighing 20 lbs (9.1 kg).

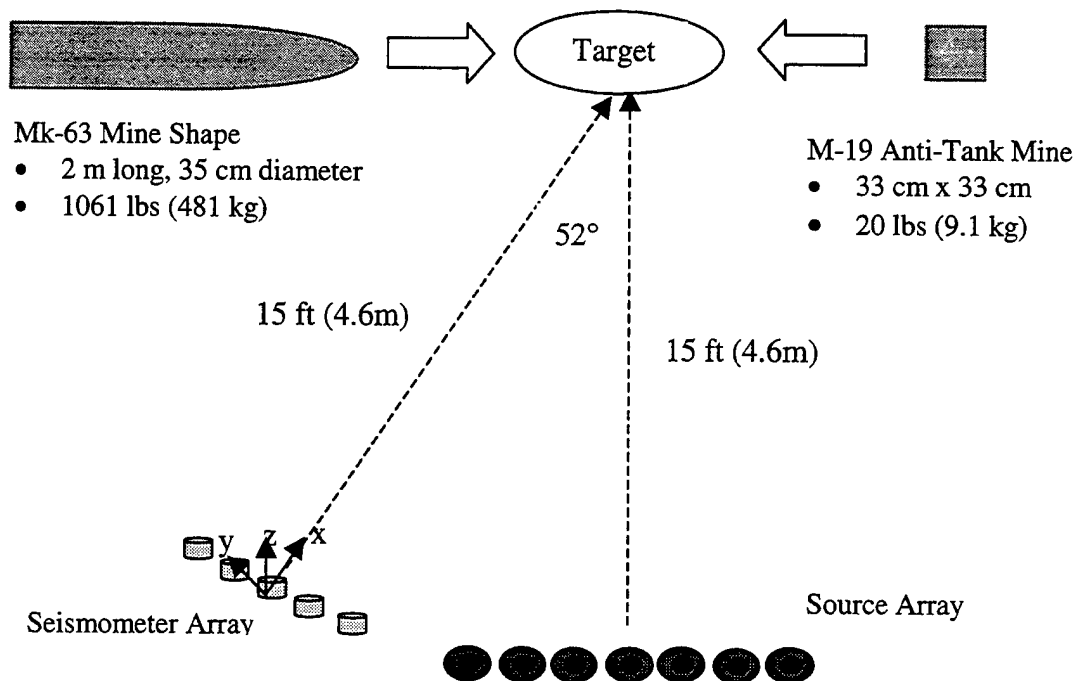


Figure 2.12 Bi-static experiment configuration

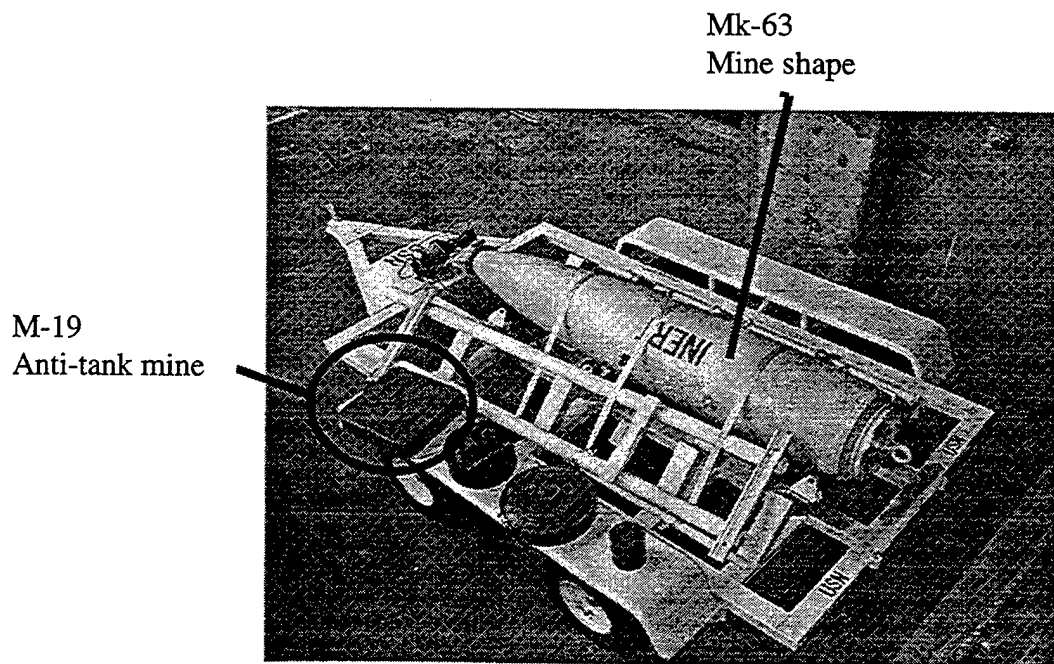


Figure 2.13. Sample Targets

All the targets were buried just so the top surface was even with the neighboring sand. In the bi-static configuration shown in Figure 2.12, the target is placed at a distance of 4.6 m (15 ft) from the sources, requiring that the seismic waves make a total roundtrip of 10.2 m (30 ft), significantly lowering the power of the received signal, as will be shown in an upcoming section. Each individual seismometer was recorded one at a time due to limitations imposed by the current data acquisition equipment. Before any targets were buried, a data set was recorded with no target present to get a baseline signal, which will be used to subtract out the direct blast and any noise. This concept, called "coherent subtraction" will be described in the following section.

THIS PAGE INTENTIONALLY LEFT BLANK

III. SIGNAL PREPARATION AND DETECTION

This section focuses on the raw data processing and methods used for detection. Some methods considered for automatic detection included methods of evaluating the time-varying frequency content of the signal for harmonics, looking at the bandwidth, correlation techniques, or using the statistics of the signal, but those proved not to work reliably. The best method for automatic detection ended up being a simple combination of a short-term energy and zero-crossing detector.

A. DATA PREPARATION

1. Array Processing

The first step in signal processing was to use a simple array technique, adding the recorded time traces using MATLAB in the following manner:

$$v_i(t) = \sum_{n=1}^m v_{sn}(t), \quad (3.1)$$

where $v_i(t)$ is the total "beam formed" signal, $v_{sn}(t)$ is the signal from the n^{th} seismometer, and m is the total number of seismometers. Array processing compacts 15 recorded signals (5 seismometers with 3 components each) into 3 x, y, and z components. This technique amplifies the on-axis signals that have the least seismic wave path difference and eliminates the off-axis signals, which have greater path differences leading to positive destructive interference [Ref. 12].

2. Coherent Subtraction

Coherent subtraction is the subtraction of a recorded signal with no target from a recorded signal with a target, leaving only information about the target. In MATLAB, it can be calculated as:

$$v_{i/CS}(t) = v_{i/T}(t) - v_{i/NT}(t), \quad i = x, y, z, \quad (3.2)$$

where $v_{iCS}(t)$ is the coherent subtracted signal for each of the three components of the seismometer, $v_{iT}(t)$ is the signal component when the target is present, and $v_{iNT}(t)$ is without a target.

In an actual unknown environment, one would not have the ability to record a clean signal with no target with 100% certainty. However, more often than not a target will not be present. If one is present in the initial recording, performing coherent subtraction without moving the equipment would cause the target signal to be eliminated. Moving the array forward or backward in small increments would eliminate the cancellation and allow the target to be detected.

3. Vector Polarization Filtering

The recordings from all five seismometers were combined into a single record consisting of an x, y, and z signal after array processing and coherent subtraction are performed. The first step in the vector polarization process is to form the radial and vertical components from x, y, and z signals recorded from the seismometer. The radial component is created from the x and y signals in the following manner:

$$\begin{aligned} r(t, \theta) &= x(t) \cos(\theta) + y(t) \sin(\theta), \\ v(t) &= z(t), \end{aligned} \tag{3.3}$$

where θ is the looking angle as described in Chapter II, Section B.3. In our experimental setup, the seismometers are aligned so θ is zero. Thus the radial component consists of the x component solely.

The next step is to apply the Hilbert transform to the radial and vertical components to produce a complex signal that contains phase information. The complex power is then computed by:

$$P_{r,v}(t) = V_r^*(t) \times V_v(t), \tag{3.4}$$

where $P_{r,v}(t)$ is the complex power, $V_{r(t)}$ and $V_{v(t)}$ are the Hilbert transformed radial and vertical components, respectively and $*$ denotes the complex conjugation operation. At this point, one can determine if any Rayleigh waves have been recorded.

The following example is illustrative of the process and will demonstrate how the Rayleigh waves show up in the imaginary part of the computed complex power. Assume the radial and vertical components, $r(t)$ and $v(t)$ are two sinusoidal signals with different phases:

$$\begin{aligned} r(t) &= A_0 \cos(2\pi\nu_0 t) \\ v(t) &= A_1 \cos(2\pi\nu_0 t + \phi). \end{aligned}$$

The following results are obtained for $r(t)$ and $v(t)$ if only the positive frequencies are used and the Hilbert transform is applied,:

$$\begin{aligned} r_{Hilbert}(t) &= A_0 e^{i2\pi\nu_0 t} = A_0 [\cos(2\pi\nu_0 t) + i \sin(2\pi\nu_0 t)] \\ v_{Hilbert}(t) &= A_1 e^{i2\pi\nu_0 t} e^{i\phi} = A_1 e^{i(2\pi\nu_0 t + \phi)} = A_1 [\cos(2\pi\nu_0 t + \phi) + i \sin(2\pi\nu_0 t + \phi)] \end{aligned}$$

Now, the complex power, $P_{rv}(t)$ can be calculated as

$$\underline{P}_{rv} = A_0 e^{-i2\pi\nu_0 t} * A_1 e^{i(2\pi\nu_0 t + \phi)} = A_0 A_1 e^{i\phi}.$$

Thus, the signal $P_{rv}(t)$ is a purely imaginary quantity due to a phase shift in between vertical and radial components. Note that the complex power is a purely real number when only body waves (P and S-waves) are present. Another characteristic of $P_{rv}(t)$ is the imaginary component of the complex power deflects positively or negatively for polarized signals according to the direction of rotation of the particle, prograde or retrograde. As a result, only the imaginary part of the complex power is plotted to determine if a target is present.

The concept of vector polarization filtering is further illustrated with the following simulation. Two signals shown in the top plot Figure 3.1 represent the radial and vertical components of a simulated signal. They are in phase, with the exception of 150 samples from index 151 to 301 when they shift 90 degrees out of phase. This is analogous to a real world situation where only body waves with no phase shift are recorded up to a point, after which Rayleigh waves are recorded. The ending of the phase shift at index 301 indicates the passing of the Rayleigh waves. The two lower plots in Figure 3.1 show the imaginary and real parts of the complex power. The plot of the imaginary part of the complex power confirms the induced phase shift between the simulated radial and vertical waves results non-zero values from time indices 151-300

and zero or close to zero values elsewhere when the waves are in phase. In this example, the particle rotation would be prograde because the imaginary complex power deflects positively, which is what one would expect from a returned surface wave. The plot of the real part of the complex power confirms that when a 90° phase shift exists, the real part is zero. In real-world signals, a mixture of body waves tend to be present throughout the duration signal, causing the real part of the complex power to have a level value. Thus, in the real part of the complex power for real-world signals there is differentiation between target and no-target and only the imaginary part is used.

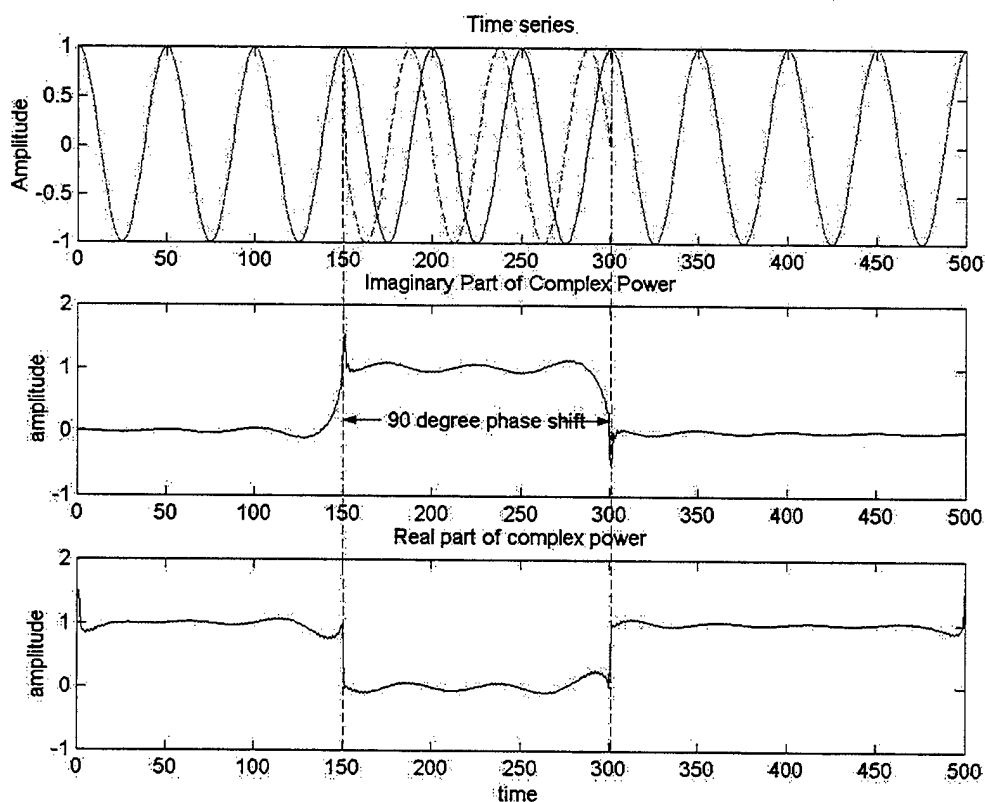


Figure 3.1. Illustration of vector polarization processing to extract the complex power from two sinusoidal figures representing radial and vertical Rayleigh waves.

4. Results of Coherent Subtraction and Vector Polarization

The top two plots of Figure 3.2 show the radial and vertical components of particle velocity as a function of time with and without the Mk-63 mine shape which was buried at a range of 15 ft (4.6m) and the bottom trace shows the imaginary component of

the complex power. An incident wave that arrives directly from the seismic source array and lasts about 0.09 seconds is observed. It is present in all signals, imposing a “minimum range” for the seismic sonar. For Rayleigh waves with an estimated speed of 75 m/s, this corresponds to a minimum range of approximately 13 ft (4m).

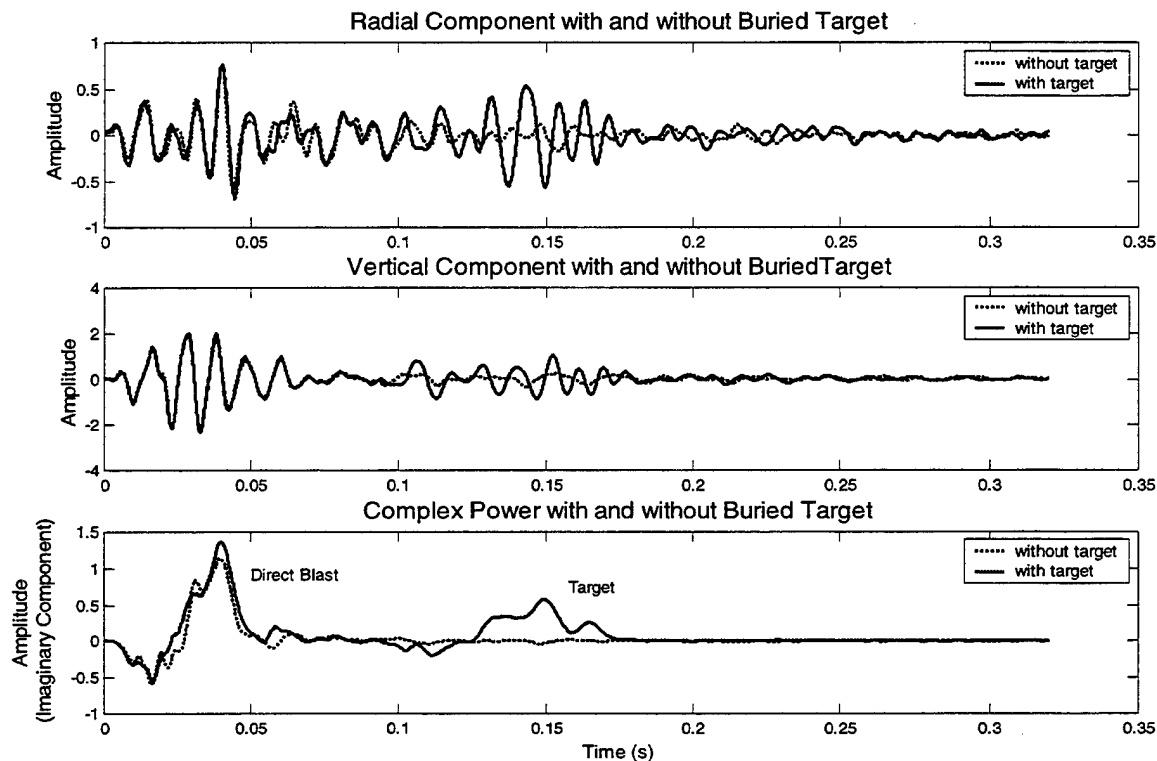


Figure 3.2. Target Data for Mk-63 Bomb before coherent subtraction

Coherent subtraction is illustrated in Figure 3.3 using the M-19 anti-tank mine signal as an example. The imaginary complex power is plotted before coherent subtraction is performed and then after. As stated earlier, the radial and vertical components are coherently subtracted before the complex power is calculated. Sheetz calculated a 9 dB RMS signal to noise ratio (SNR) for the M-19 mine and a 21 dB RMS SNR for the Mk-63 bomb, a significant improvement numerically and visually in target detection [Ref. 12].

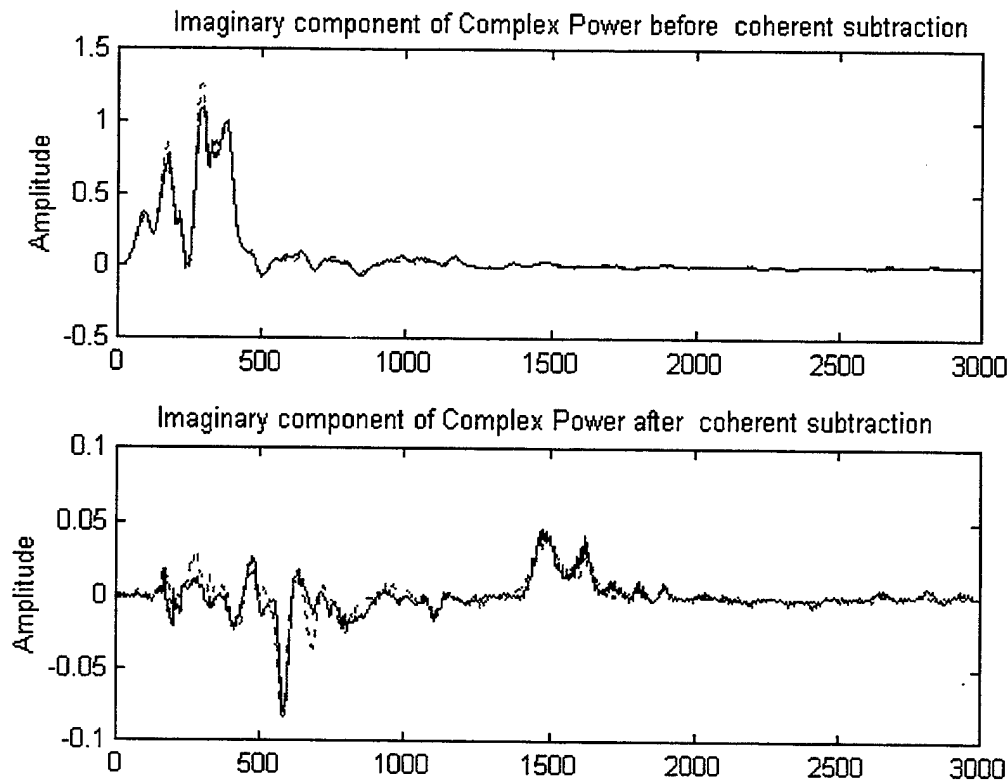


Figure 3.3. Mk-19 Anti-tank mine imaginary complex power before and after coherent subtraction is performed.

B. DETECTION

Significant effort was put into developing a simple and computationally cheap routine that could automatically determine if a target was present in a given signal and extract the relevant section for further classification. Having this ability would greatly speed up the mine clearance process that the Navy desires to complete in a very rapid manner, as stated in the ORD [Ref. 5].

Several methods, including time-frequency content and harmonic resonance, higher order moments (kurtosis and skewness), and bandwidth determination of signals were investigated to extract a target signal from the imaginary complex power signal before coherent subtraction was performed. Some of these methods only worked on the largest of the signal while others did not work at all for an easily automated scheme. In

the end, it was determined that using a coherently subtracted signal was a viable option as it has merit for future development and implementation in a practical system. Using the coherent subtracted signal allowed a very basic and efficient scheme to be implemented, as discussed below.

1. Short-term Energy and Zero-crossing Rate Detector

Note that a target signal is easy to see visually on a plot after performing array processing, coherence subtraction, and vector polarization filtering. A computationally cheap and fast scheme had to be developed to first determine if a target was present and then to extract it for classification. All targets signals exhibit some similar properties in the time traces that can be taken advantage of: 1) when a target is present, the majority of energy in the signal is contained in the portion where target return is present, 2) when no target is present, the signal is noise-like and fluctuates rapidly around zero. Short-term energy and zero-crossing rates of the signal are calculated and plotted to differentiate between target/no-target signals. The short-term energy routine amplifies the portion of the signal where energy is present and decreases the lower, noise like levels. The zero-crossing rate routine is complimentary to the short-term energy, exaggerating the noise-like parts of the signal and forcing higher energy portions to zero.

The short-term energy and zero-crossing routines in MATLAB both have user specified window frame lengths and overlap percentages. The short-term energy is calculated in the portion of the signal covered by the frame by summing the squares of each sample. The zero-crossing rate is a measure of how many times the signal changes signs in the portion of the signal covered by the current frame. Successive frames overlap by the percentage specified by the user. Values for window length and percentage overlap used (25 samples and 50% overlap) were determined by trial and error to give the best generic results for all classes of signals. (See Appendix A for code)

Figure 3.4 illustrates the detection and extraction process using short-term energy and zero-crossing calculations. The mean of the signal is not removed prior to the detection algorithm because the no-target portion of the signal is naturally close to zero-mean, allowing the zero-crossing algorithm to emphasize the noise-like fluctuation

around zero. If a bias was introduced into the system from equipment or an external source, it would have to be removed, but this condition did not exist in our experiments. A combination of two threshold detectors was used to detect the beginning of the target return signal. A value above the specified threshold for the short-term energy combined with 25 successive zero-crossing values that were zero indicated that a target signal was present. The beginning time index was saved and the ending time index was sought, indicated by the short-term energy falling below a specified threshold and zero-crossing rates increasing above zero. The starting and ending time indices are used to extract the portion of the imaginary complex power signal that contains the target. The threshold value and the number of zero-crossing samples to average together were determined by trial and error until the detection algorithm was able to pull out the target signal.

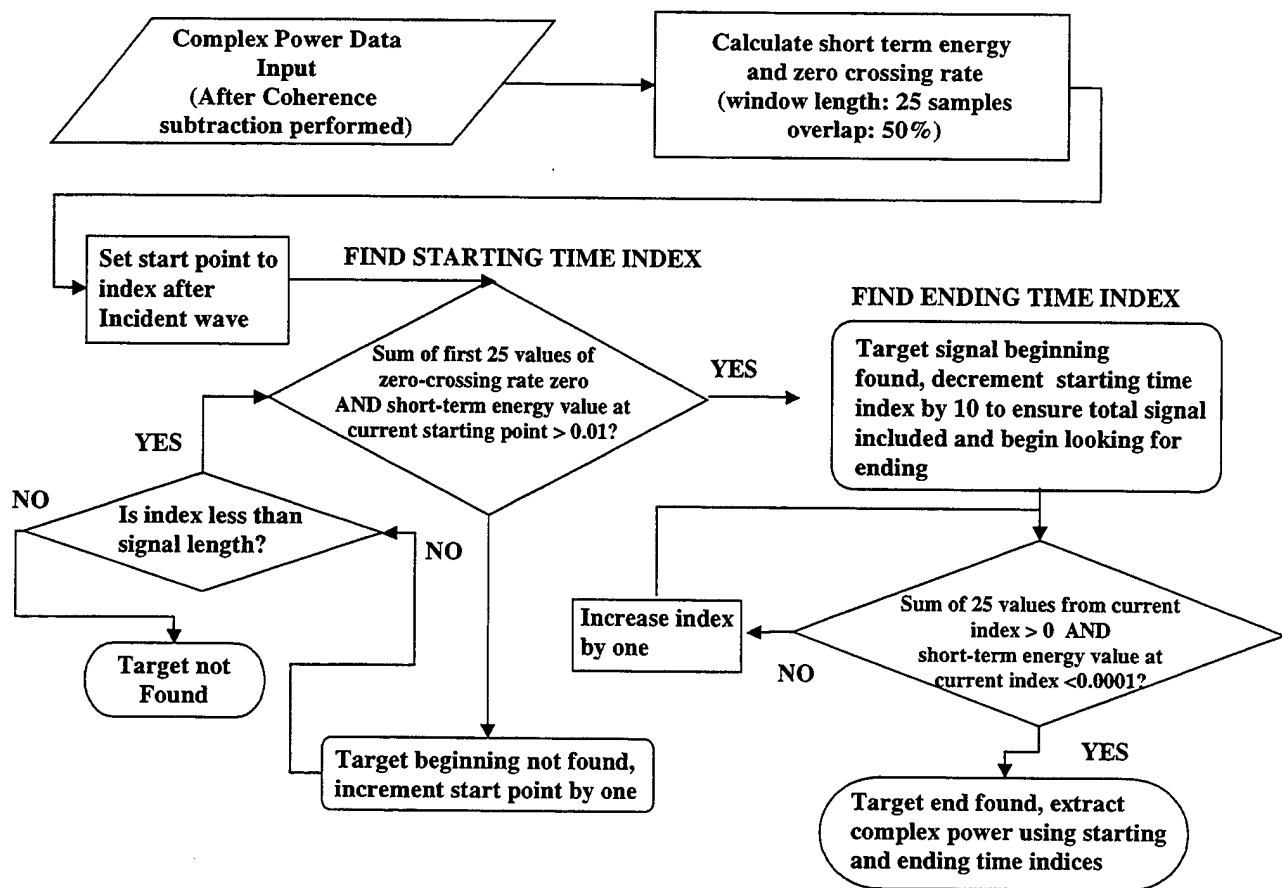


Figure 3.4. Flow chart for short-term energy and zero-crossing rate detector

2. Results/Extracted Target Signals

The extracted signals are shown in Figure 3.5 with multiple trials for each target type. The short-term energy and zero-crossing rate detector was able to correctly identify and extract targets signals from the imaginary coherent subtracted complex power signals and reject signals recorded without a target. Experiments conducted using a concrete man-made rock-like object found on the Navy Beach in Monterey were an exception to the 100% detection rate of the algorithm. Of the three recordings conducted using the rock as a target, only one was identified as a potential target using the automatic detection algorithm presented in this chapter.

From a visual inspection, all of the signals seem to have a unique shape and reasonably similar results for multiple trials. One trial of the gas cylinder seems to be significantly different than the other two and could possibly be an outlier or exception to the normal response generated from that type of target. The anti-tank mine and the scuba tank appear similar but are slightly time delayed, thus any feature extraction method would have to be time-invariant.

C. FAILED METHODS

1. Bandwidth Determination

After performing array processing, several observations regarding the behavior of the signals were made. All signals are low frequency due to the nature of the seismic wave and the source generating them. From visual inspections of the power spectral densities of the signals, target signals appeared to be narrowband compared to no-target signals, which are more wideband in nature as shown in Figure 3.6. Using a narrowband or wideband signal discriminator to detect a signal for sonar applications has been proposed and used before with success in an expert classifier system [Ref. 16]. An automatic target detection algorithm that would discriminate between wideband and narrowband signals was implemented by calculating the ratio of power contained in the first 50 Hz of the signal of compared to the overall signal. For the largest target, the

MK-63 mine shape, close to 95% of the energy lie in this region, indicating a narrow band target signal was present. Unfortunately, similar results were not obtained with the remainder of the targets, with only approximately 50% of the power residing in this region, making detection at best a 50% guess at best.

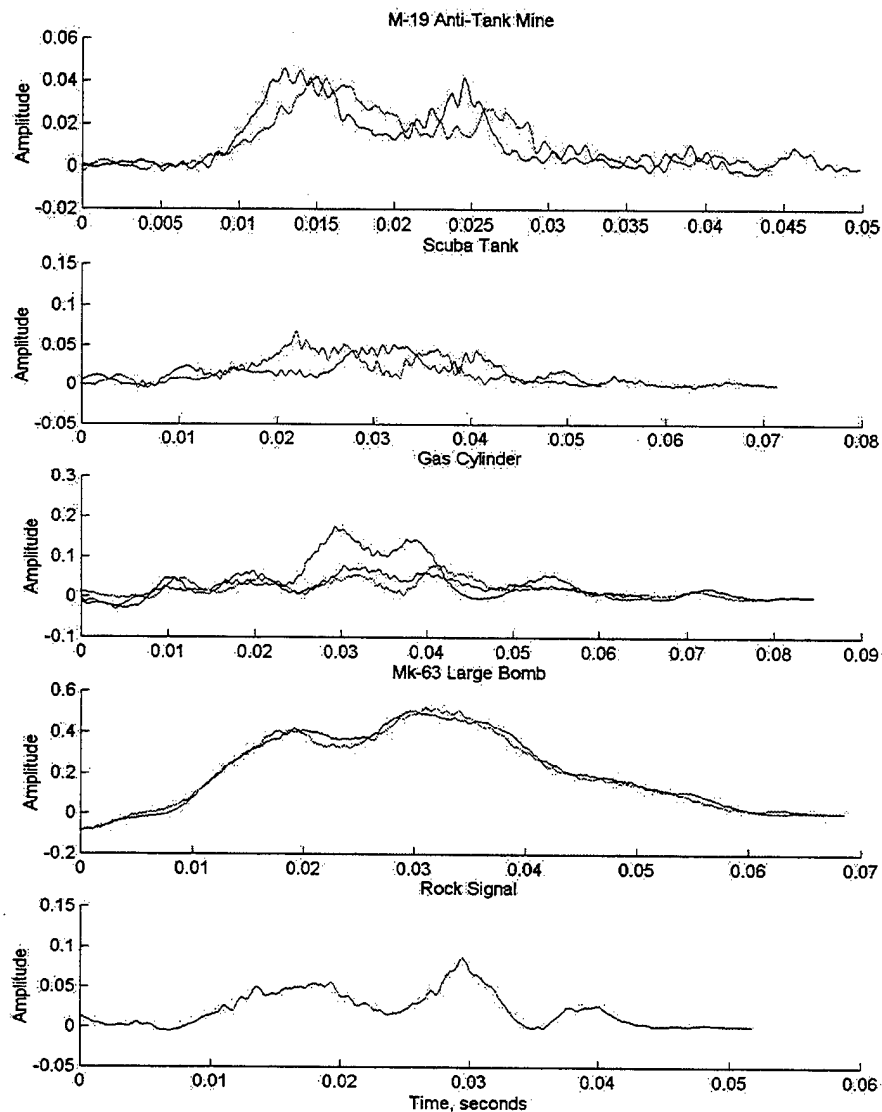


Figure 3.5. Extracted target signals from imaginary complex power data for multiple trials.

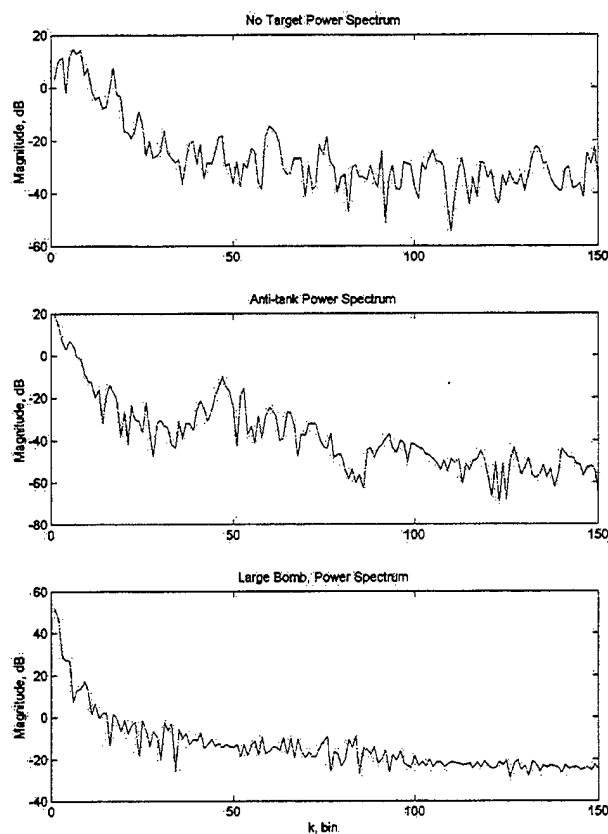


Figure 3.6. Frequency spectrum of a no-target, anti-tank, and large bomb signal used in a bandwidth determination detection scheme.

2. Spectrograms and Harmonic Correlation

After the initial signal processing, the spectrogram, or time-varying frequency content of the imaginary complex power was calculated and plotted, as shown in the top plot of Figure 3.7. The incident wave clearly shows up as the presence of energy in a large portion of the frequencies, along with what appeared to be harmonics due to the triangular-like shape of the outgoing pulse. When a target was present it appeared that harmonics were being reflected off the object and recorded. A detection scheme was implemented to measure the amount of correlation between the outgoing harmonics and what was returned. If there was a high amount of correlation, this could be interpreted as an indication of the presence of a target. The correlation was normalized by

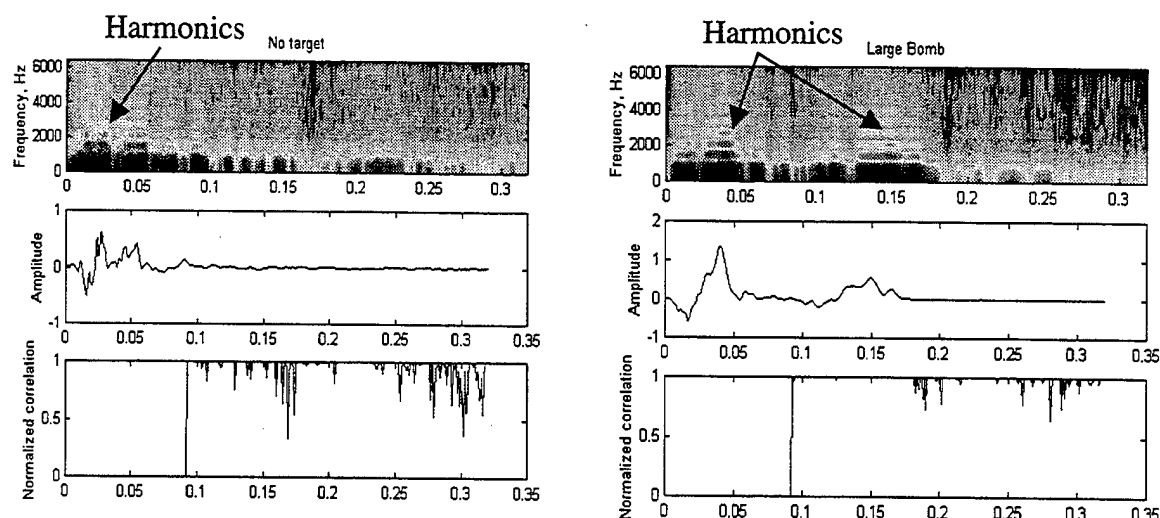


Figure 3.7. Spectrograms of imaginary complex power, time series of data, and correlation between outgoing harmonic reference and recorded data.

the energy contained in the reference signal and the test signal to ensure that only the similarity between harmonics was being measured. When these quantities were calculated and plotted, as shown in the bottom plots of Figure 3.7, no meaningful results were obtained. The amplitude of the correlation between the reference harmonic signal, taken from the outgoing wave and the remaining data from the spectrogram, does not vary significantly from one. This indicates the presence of target is not dependent on returned harmonics, but the magnitude of the energy returned. A possibility also exists that the harmonics are an artificiality of the windowing used in processing the data. For these reasons, harmonic correlation of spectrogram data was ruled out as a viable method for automatic detection.

3. Kurtosis and Skewness

Based on an article that successfully used higher-order statistics to detect the onset of P and S waves in seismic events by looking at the short time kurtosis and skewness of a signal [Ref. 17], a similar method was implemented using a short-term kurtosis and skewness measurements of the imaginary part of the complex power. The signal was windowed in small segments, calculating the two higher-order statistics of

interests, then moving the window over the signal allowing for overlap. Any large change in the kurtosis and skewness would indicate a change in the distribution of the

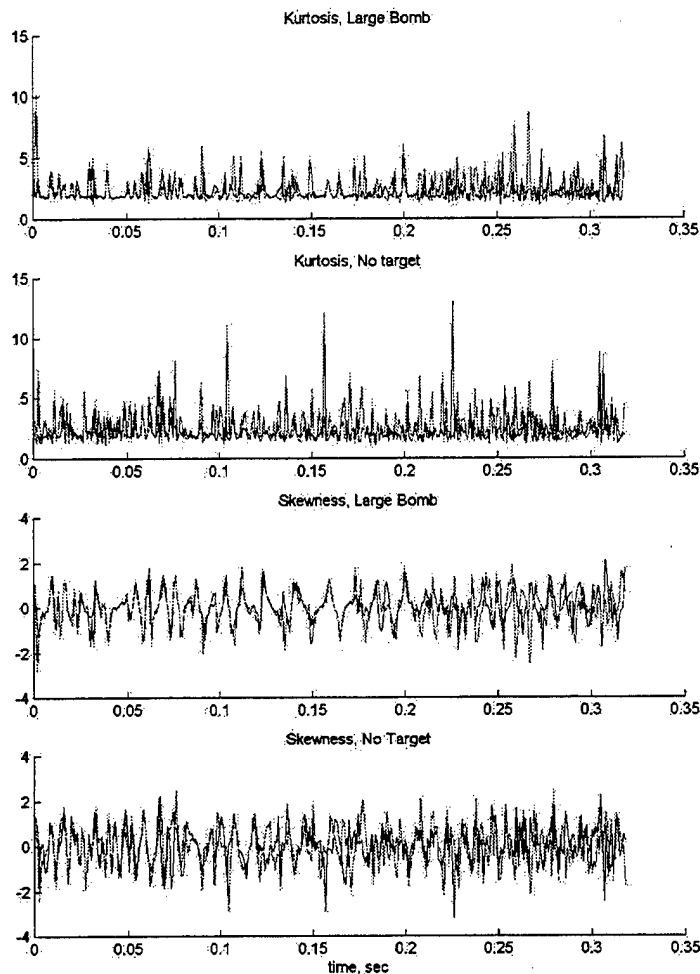


Figure 3.8. Kurtosis and Skewness measurements for multiple trials of a large bomb and no target.

data. Unfortunately, as the results in Figure 3.8 show, the data does not have enough of a change in distribution from a target and no-target portion of the signal to be of use in an automatic detection scheme.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. FEATURE EXTRACTION AND CLASSIFICATION

In the classification problem, there are three basic steps: low-dimensional signal characterization (data projection), feature extraction and optimization, and construction of classifier topologies [Ref. 18]. The techniques presented in Chapter III can be thought of as a form of data projection, using array processing, coherent subtraction and vector polarization to enhance the signals and make features easier to extract. This section deals with the last two steps in signal classification, feature extraction and classifiers.

Feature extraction and classification of signals is a thoroughly developed area of modern signal processing with an abundance of literature on the topic, but methods that have been applied directly to seismic data of the nature presented in this paper have not been fully explored. This section discusses general methods of classification, past methods used, and the steps involved in developing an automatic classification system, focusing on feature extraction. Three methods of extracting features of signals are presented, followed by a basic description of how these features would be used in a computationally simple classification scheme.

A. CLASSIFICATION METHODS

1. Feature Extraction

The goal of feature extraction is to extract information about each class of signal that is capable of discriminating it from other classes of data. A key attribute of the signal is computed and collected in a vector format, which can be considered as a type of data compression that removes irrelevant information and preserves the relevant information. Good features would include the following attributes:

1. large interclass mean and distance and small intraclass variance,
2. insensitivity to extraneous variables (low SNR dependency),
3. computationally inexpensive to measure,
4. uncorrelated with other features,

5. mathematically definable, and
6. explainable in physical terms [Ref. 18].

In addition, some more mathematical properties have been proposed for desired feature attributes that include: 1) invariance to time-shifts, 2) invariance to time-scalings due to different propagation speeds in the medium, 3) insensitivity to multiple reflections, and 4) insensitivity to additive noise [Ref. 19].

Common feature extraction methods include the Fourier Spectrum, Fourier-Mellin transform, cepstrum, bispectrum, discrete wavelet transform, pole estimation, histogram estimation, and time samples [Ref. 19]. A feature can be time-domain raw data, but it is not desirable because of a lack of data reduction, sensitivity to noise and other interference, and an exponential increase in data required for training as a function of the feature dimension [Ref. 18]. A large majority of these methods were investigated for the signals presented in this thesis, but did not exhibit any unique characteristics between classes.

The effectiveness of a feature set is determined by how well different classes can be separated. Given a statistically significant number of trials (usually more than 20), decision boundaries that separate the features are established, allowing decisions to be made with respect to class membership. These boundaries are determined by the probability distributions of the patterns belonging to each class, which must be specified or learned [Ref. 20]. In this thesis, the patterns and decision boundaries are not known and will have to be learned from experimental data.

2. General Classification and Pattern Recognition Methods

Classification architectures used in the mine detection problem fall into several main categories: hypothesis testing, pattern recognition, and Hidden Markov Models (HMM). Most of the data evaluated using these methods is obtained from sources, such as Ground Penetrating Radar (GPR) or various types of imagery which have a much wider bandwidth and contain more information in the signal available for feature

extraction as compared to the very low frequency, low bandwidth seismic signal investigated in this thesis [Ref. 19].

Hypothesis testing involves comparing the *a-posterior* probabilities so that if one is given a measurement, and two classes to decide on, c_1 and c_2 , the goal is to define the decision statistic so that

$$y \in c_i \text{ if } p(c_i | y) > p(c_j | y), i \neq j. \quad (4.1)$$

The *a-posterior* probabilities in eq (4.1) are usually difficult to calculate, so the Bayes theorem is used to to rewrite the *a-posterior* probability as:

$$p(c_i | y) = \frac{p(y | c_i)p(c_i)}{p(y)}. \quad (4.2)$$

Replacing eq (4.2) in (4.1) leads to:

$$\frac{p(y | c_i)}{p(y | c_2)} > \frac{p(c_2)}{p(c_i)} = \gamma. \quad (4.3)$$

This method of classification can be computationally intensive and will not be the preferred method in the approach of this thesis.

Pattern recognition classifiers are usually separated into two different types: conventional and artificial neural networks. Conventional methods include Multivariate Gaussian (MVG), Gaussian Mixture Models (GMM), k-nearest neighbor (KNN), Learning Vector Quantizer (LVQ), and Binary and Polynomial Tree Classifiers (BTC/PTC). Artificial Neural Networks include Probabalistic Neural Networks (PNN), Back Propagation Network (BPN), and discriminant Neural Networks (DNN) [Ref. 18].

Hidden Markov Models (HMMs) are a very powerful technique that models the temporal structure of variability of a signal. The HMM theory is based on the Markov Chain, a probabilistic description of transitions between a system's states [Ref. 18]. This method has been very successful in such fields as speech processing, human face identification, optical character recognition, and DNA modeling [Ref. 21]. Conceptually,

they are very hard to analyze, require a significant amount of data for training, and are computationally expensive.

In earlier work, Zambartas applied HMMs and neural networks to the seismic-sonar classification problem from data generated by a different, but very similar method to one presented in this thesis [Ref. 21]. No consistent features were found from the data provided, so time-domain information was used directly on a very limited data set. Back-propagation feed-forward neural networks (BPNN) were also implemented and tested on the data. Both classifiers performed at a 97% classification rate.

Gaussian Mixture Models (GMM) were selected as the next category of classifier to investigate due to their computational simplicity and widespread and proven use to classify land mines using other types of data. The mechanics of GMMs are explained in Section C of this chapter, but the key concept of GMMs relies on features that map to separate and distinguishable areas in a multi-dimensional space.

B. EVALUATED FEATURE EXTRACTION METHODS

When evaluating the results of different feature extraction methods, it is important to confirm that what is being observed is an actual feature of the class and not a random occurrence. A large number of trials under the same experimental conditions that yields similar features will ensure this. The data available from experiments conducted by Sheetz and Guy [Refs. 12, 13], and aided by the author, yielded only two to three trials per class, making statistically significant observations about features and feature clustering difficult. The following section describes several feature extraction methods evaluated which appear to yield similar features for the very limited number of trials available for testing. In order to verify the validity of these feature extraction methods, many more trials, on the order of 20 or more, should be used to see if similar results are obtained.

1. Higher Order Moments

The central moment of order k of a distribution is defined as

$$m_n = E(x - \mu)^k \quad (4.4)$$

where $E(x)$ is the expected value of x . The first central moment is zero and the second central moment is the variance, or power, of the signal. These moments are most commonly used to summarize the probability density function (pdf) of a random variable. Under certain conditions, a pdf is can be completely specified if all the expected powers of x , or the moments, are known [Ref. 22].

Other parameters are available to characterize the pdf of a variable, such as the kurtosis and skewness. The kurtosis is a measure of how outlier-prone a distribution is and is defined as

$$k = \frac{E(x - \mu)^4}{\sigma^4}, \quad (4.5)$$

where μ is the mean of x , σ is the standard deviation of x , and $E(t)$ represents the expected value of the quantity t . The kurtosis of the normal distribution is 3. Distributions that are more outlier-prone than the normal distribution have kurtosis greater than 3; distributions that are less outlier-prone have kurtosis less than 3. The skewness is a measure of the degree of asymmetry of the pdf about its mean and is defined as

$$y = \frac{E(x - \mu)^3}{\sigma^3}. \quad (4.6)$$

It can be shown that the skewness is zero when the pdf is symmetric about its mean [Ref. 22]. Calculating higher order moments, to include the kurtosis and skewness, reduces the input data into a significantly smaller number of data points that could be used as features if the distribution of the data is different for each class.

The detection algorithm developed in Chapter II does not extract signals of equal length, as it makes no assumptions about the duration of a return signal. Longer signals will contain more energy because there are more data points, so signals from the same class could vary based solely on their lengths. As a result, the signals were limited to the

shortest signal in all of the classes to avoid this situation. The mean of the signals was removed and the signals were normalized to contain unit energy before computing higher order moments.

The higher order moments were plotted in various arrangements to see what produced the best clustering for the classes. The best results are shown in Figures 4.1, 4.2, and 4.3. Although there are only two to three trials per class of target, it would appear that this feature extraction method may hold promise for use in a classifier. The observed clusters for each class seem to demonstrate the desired property of large interclass means and distances and small intraclass variances. Additionally, the time invariance property is satisfied as the pdf, which the higher order moments represent in some sense, is not effect by a time shift in a signal. The anti-tank and scuba tank clusters seem to be the closest to each other with a possibility of some overlap. The scuba tank has one outlier that is extremely different from the other two trials and maybe an exception to the norm for this case. More trials are needed for each class to see if the clustering trend based on higher order moments continues.

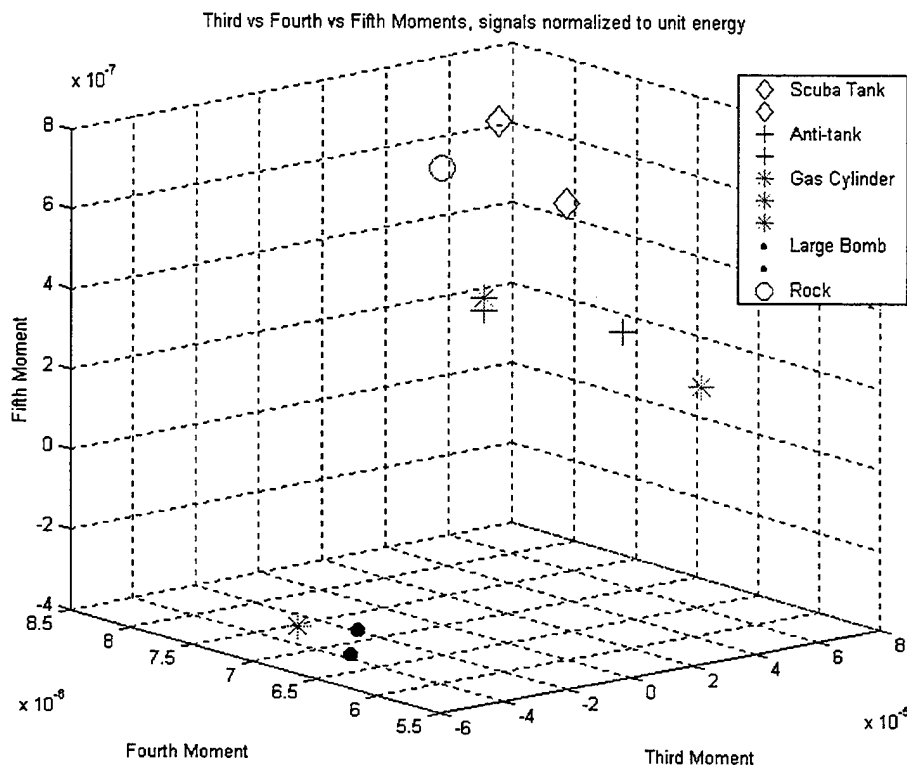


Figure 4.1. Clustering of features based on the third, fourth, and fifth moment.

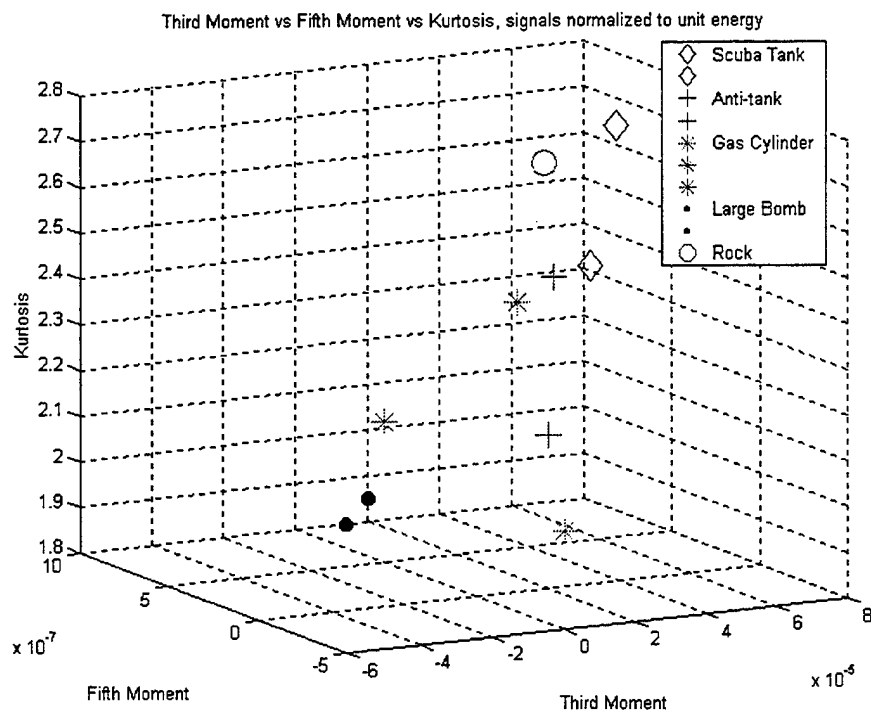


Figure 4.2. Clustering of features based on the third moment, fifth moment, and the Kurtosis.

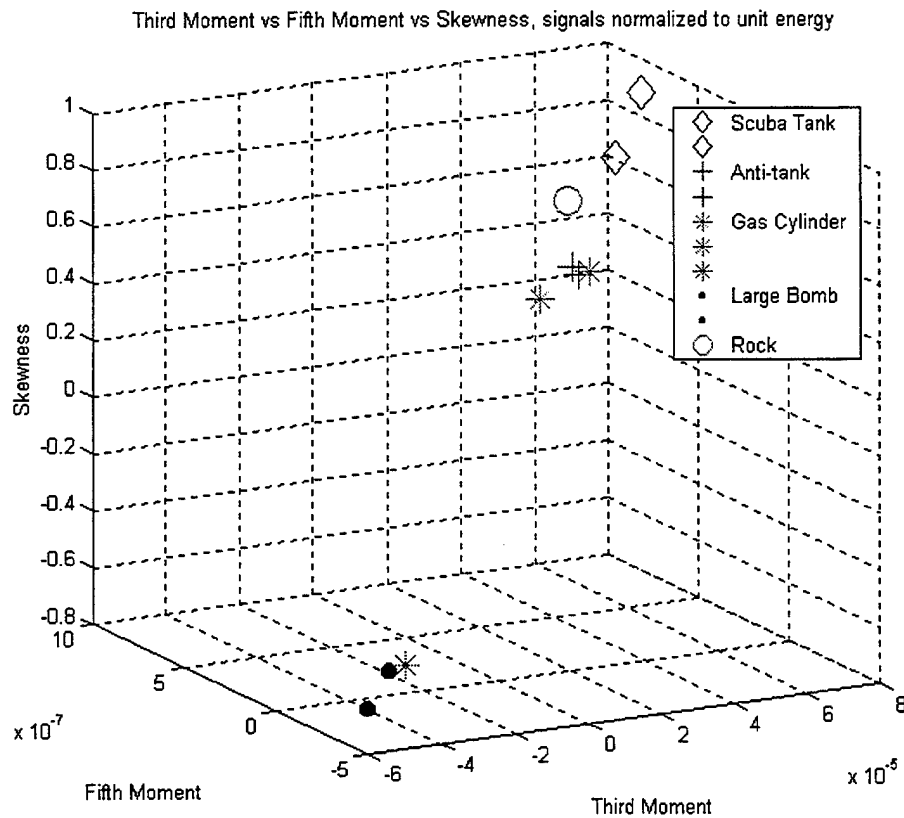


Figure 4.3. Clustering of features based on the third moment, fifth moment, and the Skewness.

2. Impulse Response Modeling Using STMCB Iteration

Many methods have been studied for modeling data as an impulse response of a filter, the most well known being AR, ARMA, and Prony's method [Ref. 23]. If the impulse response of a filter sufficiently represents a signal, then the poles and zeros of the filter could be used as features in a classifier if they cluster for each class. Several different methods were evaluated in an attempt to model the target signals as an impulse response. The best results were obtained by using the Steiglitz-McBride (STMCB) method of iterative prefiltering approach to direct modeling of the data, which is available as a MATLAB function *stmcb.m*. Given an impulse response, x , and specifying the number of poles, P , zeros, Q , and maximum number of iterations, *stmcb.m* finds the coefficients of the system $B(z)/A(z)$. For example, if $P = 2$ and $Q = 1$, then

$$\frac{B(z)}{A(z)} = \frac{b_0 + b_1 z^{-1}}{1 + a_1 z^{-1} + a_2 z^{-2}}. \quad (4.7)$$

error function is given by

$$E^{(i+1)}(z) = \frac{X(z)A^{(i+1)}(z) - B^{(i+1)}(z)}{A^{(i)}(z)}, \quad (4.8)$$

where the superscripts (i) and $(i+1)$ denoted the values of the functions at the i^{th} and $(i+1)^{\text{th}}$ iterations. The A and B terms are chosen to minimize the corresponding sum of squared errors at each iteration. For a more in depth analysis of the Steiglitz-McBride method, please consult [Ref. 23].

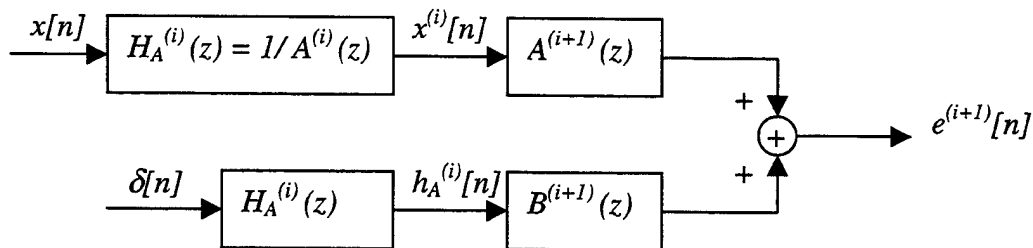


Figure 4.4. Block diagram for iterative prefiltering. From [Ref. 23].

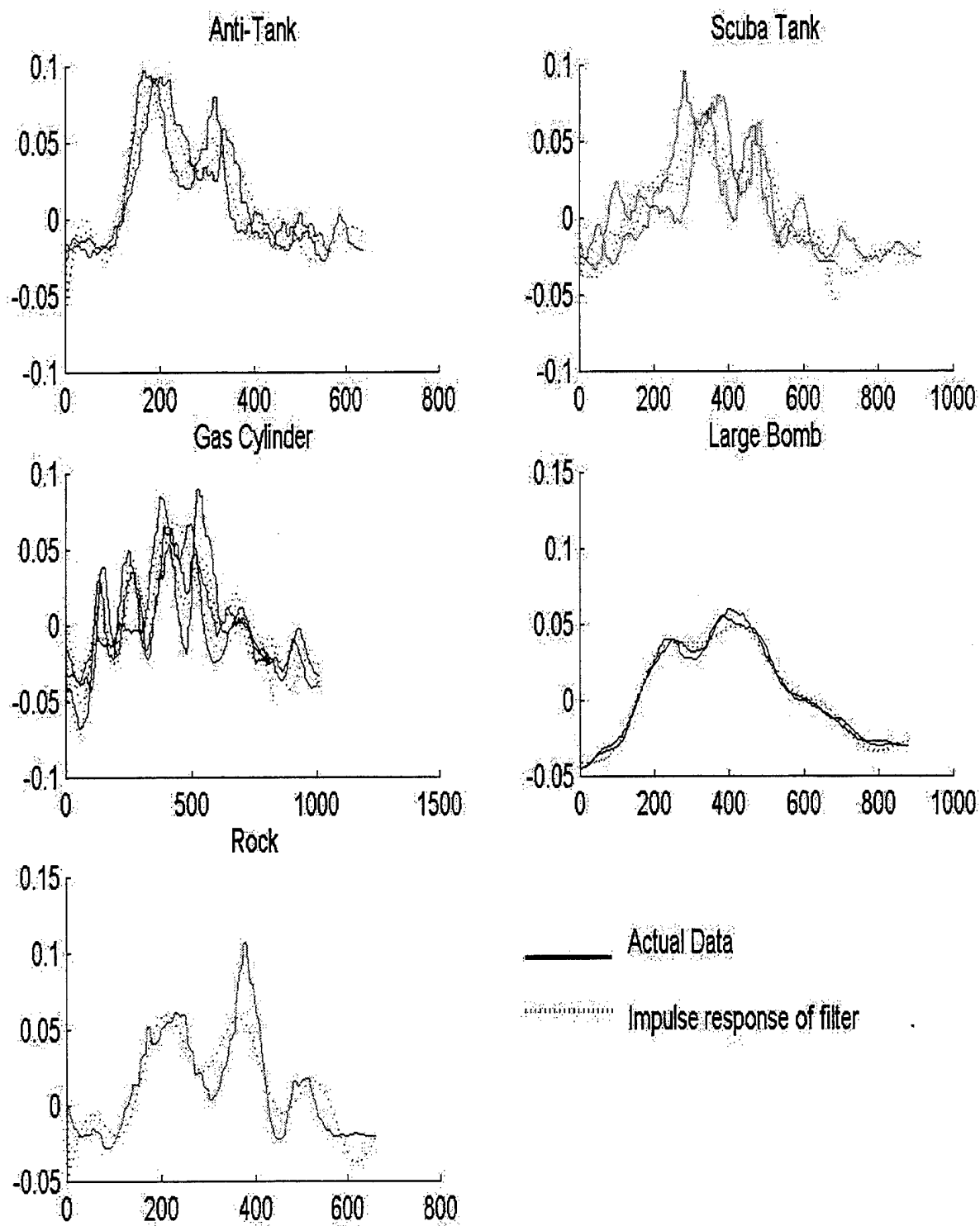


Figure 4.5. Original signals superimposed with impulse responses from filters using the *stmcb.m* iteration, 6 zeros, and 4 poles.

As illustrated in Figure 4.5, the STMCB iterative method does a reasonably good job in estimating the signal by modeling it as the impulse response of a filter. Figure 4.6 shows the pole-zero plot for all classes of targets using 6 zeros, 4 poles, and 8 iterations. The order for the transfer function was determined simply by trial and error, yielding poles in locations distinguishable by class. Using a larger number of zeros than poles, or a higher order numerator than the denominator, forces the poles to fall in a very small region all on top of each other. Therefore, the poles cannot be used as features. Again, if more trials were available, it would appear as if the zeros of the transfer function filter could be used as features.

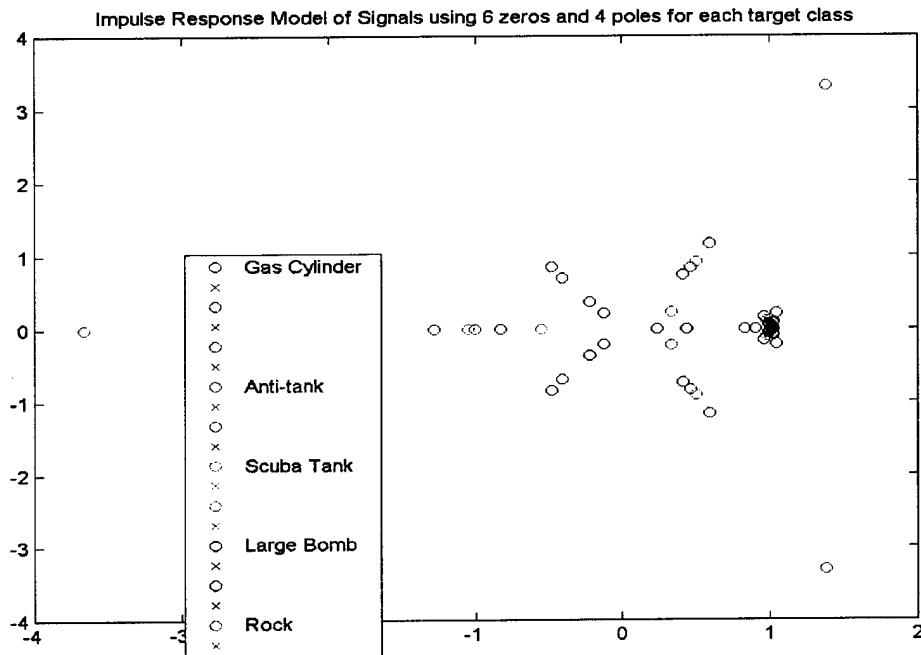


Figure 4.6. Poles and zeros of impulse response modeling using STMCB iteration with 6 zeros and 4 poles.

3. Radial Basis Function Modeling of Data

A Radial Basis Function Network (RBFN) is a linear model for a function, $t(\underline{x})$ in the form:

$$f(\underline{x}) = \sum a_i g_i(\underline{x}), \quad (4.9)$$

where a_i is a weight and $g_i(\underline{x})$ is the basis function. A radial function is defined as having a response decreasing or increasing monotonically with the distance from a central point. The most common radial basis function used is the normal or Gaussian function:

$$g(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}. \quad (4.10)$$

Representing a function with a radial basis function allows for exact interpolation of a data set, requiring every input vector to be mapped exactly into a specific target vector. Applying this approach to the target data sets, the input vector corresponds to the time index and the target vector corresponds to the signal vector extracted from the automatic detection algorithm presented in Chapter III. The signals are then represented as a weighted sum of gaussian shaped pulses and the means, standard deviations, and weights can then be evaluated to see if they are unique for each class and suitable for use as features in a classifier.

Several methods are available to automatically model the signal as a sum of weighted guassians. The MATLAB function 'newrb.m' automatically designs a RBFN that approximates the given function, but the user must specify the width of the Gaussians used to represent the signal. Widths could be experimentally determined for feature analysis with prior knowledge of the class, but this is not suitable for a field environment where the class of target is not known. The algorithm that fits the Gaussian functions to the target signal should be able to make a best fit to the input signal with no knowledge of the class type.

A NETLAB MATLAB toolbox available on the Internet provides such algorithms [Ref. 24]. The software provided in NETLAB performs simple regression using a radial basis function network. The user supplies the input data, the target data, and the number of gaussians used to approximate the target data. The toolbox designs a RBFN using a Gaussian Mixture Model (GMM) trained with the Expectation Maximization (EM) algorithm to find the centers (μ_i) of the pulses. The GMM and EM algorithms will be explained in the following section. After the centers are found, the standard deviation or

widths of each gaussian is set to be the average of the distances between the centers times a user specified scale.

Now the problem is reduced to a simple design matrix:

$$\begin{bmatrix} t_{i1} \\ \vdots \\ t_{iN} \end{bmatrix} = \begin{bmatrix} \Phi_1(\underline{x}_1) & \cdots & \Phi_M(\underline{x}_1) \\ \vdots & \ddots & \vdots \\ \Phi_1(\underline{x}_N) & \cdots & \Phi_M(\underline{x}_N) \end{bmatrix} \begin{bmatrix} w_{i1} \\ \vdots \\ w_{iN} \end{bmatrix}; \quad i=1,\dots,K \quad (4.11)$$

$$\mathbf{T} = \Phi \mathbf{W},$$

where \mathbf{T} is the target matrix, Φ is the gaussian radial basis function network defined by 4.10, the means and variances found by the GMM and the input, and \mathbf{W} are the weights. The next step is to find the weights of each Gaussian, which is accomplished by using the pseudo-inverse of the design matrix:

$$\mathbf{W} = (\Phi^T \Phi)^{-1} \Phi^T \mathbf{T}. \quad (4.12)$$

The specific functions used from the NETLAB MATLAB code to generate the results are *rbf.m*, *rbftrain.m*, and *rbffwd.m* and are included in Appendix A. For more information on their operation, please consult [Ref. 24].

Figures 4.7 and 4.9 show the results when 15 Gaussians are used to represent the data with a scale factor of 0.3. The reconstructed signals are very close approximations to the original data, as shown in Figure 4.7. Figure 4.9 shows a plot of the features that were evaluated to have the best clustering effect that could be used as features. As a reminder only two trials were available per class, so this is only an initial effort at establishing these parameters as features. The centers were spread evenly over the input range for all classes so they did not contain any information that could help distinguish between classes.

By plotting the weights of the Gaussian pulses against the width of the pulses determined by the algorithm, it appears as if the 5 classes available have distinct features. The EM algorithm is not guaranteed to converge to the same point every time it is run because of two points: 1) it depends on the initial conditions selected, which are normal random variables in the NETLAB algorithm, and 2) the algorithm converges to a local maximum, not a global maximum with no guarantee that there is more than one local

maximum [Ref. 25]. Twenty trials of the EM algorithm were run and plotted in all the clustering plots (Figures 4.8, 4.10, 4.12) to ensure that the results would be in a reasonable distance from each other and remain separable from other classes.

Similar results were obtained using 8 and 9 gaussians to represent the data. With a smaller number of gaussians the original data sequence is not modeled as well as with 15, as seen in Figures 4.9 and 4.11, but sufficiently enough to exhibit the same clustering as seen with 15 neurons. A smaller number of features is more desirable to reduce processing time and computations, but again with only two trials per class, quantitative measurements about the optimal number of gaussians to use for feature extraction cannot be made.

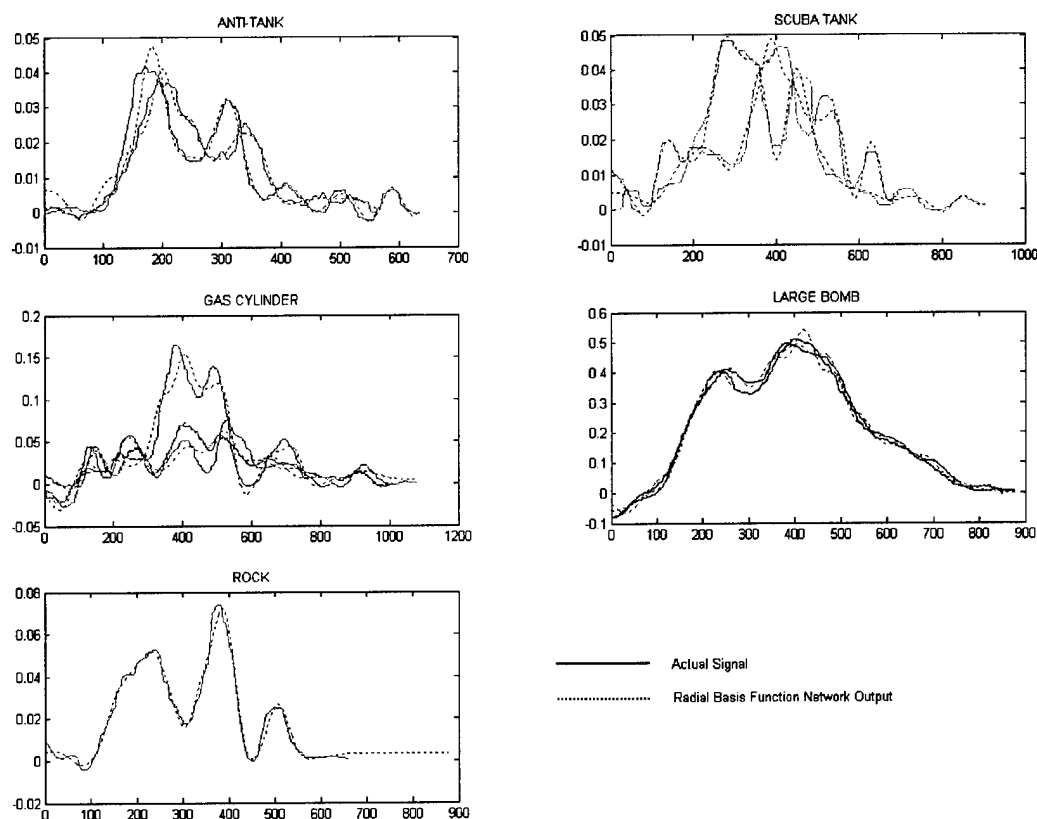


Figure 4.7. RBFN approximation of target signal using 15 hidden layers and a spread factor of 0.3.

One trial of the gas cylinder is notably different, both in the time domain and in the feature plot. Most likely this trial was an aberration and the associated features for this trial can be considered an outlier, as indicated in Figures 4.8, 4.10, and 4.12. When training a classifier, this trial can be ignored or pruned away to prevent incorrect results.

The two trials for the scuba tank do not yield results that cluster in the same manner as the other classes do. Again, one of the signals could be an outlier and could be eventually disregarded during classifier training if more trials were available. Even though the time series for the two trials of the scuba tank look similar, a larger and longer delayed second bump in one signal forces the RBFN algorithm to assign a higher spread to the signal.

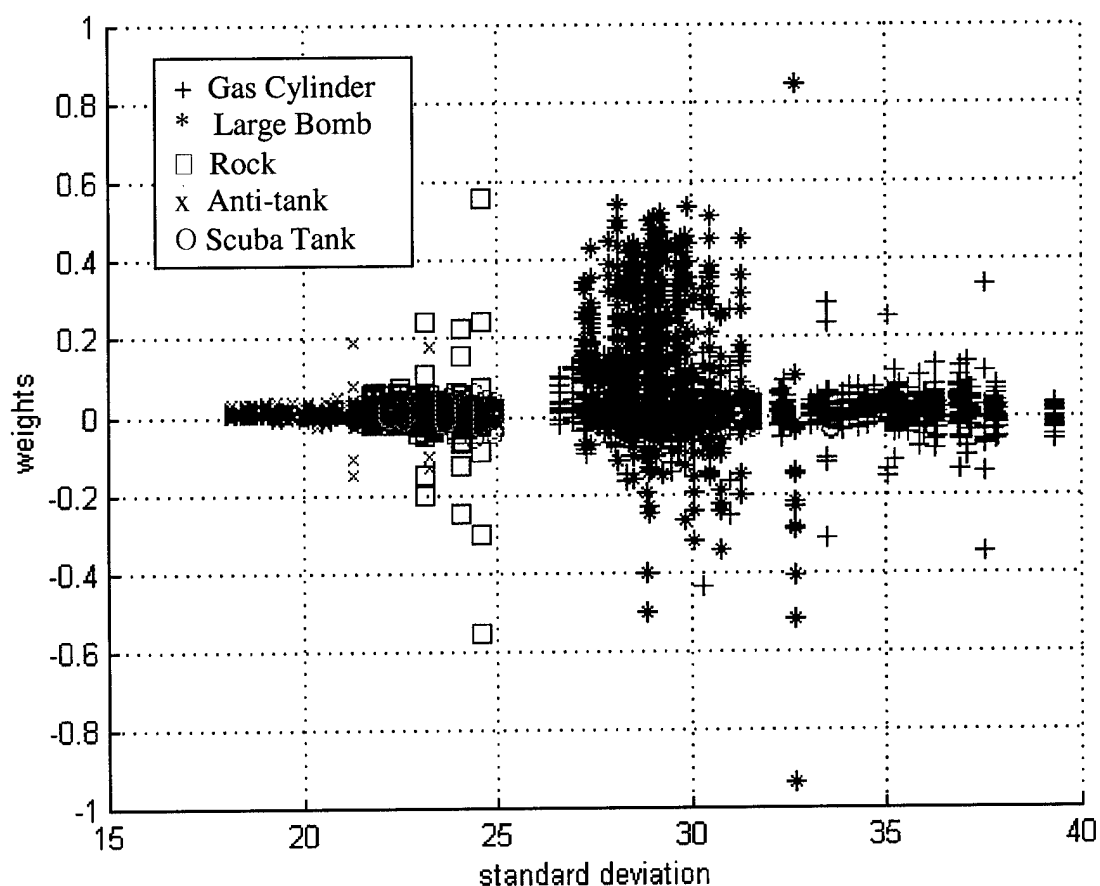


Figure 4.8. Weights vs widths of 20 trials per signal using an RBFN network generated by GMMs and the EM algorithm, 15 basis functions, spread scale = 0.3.

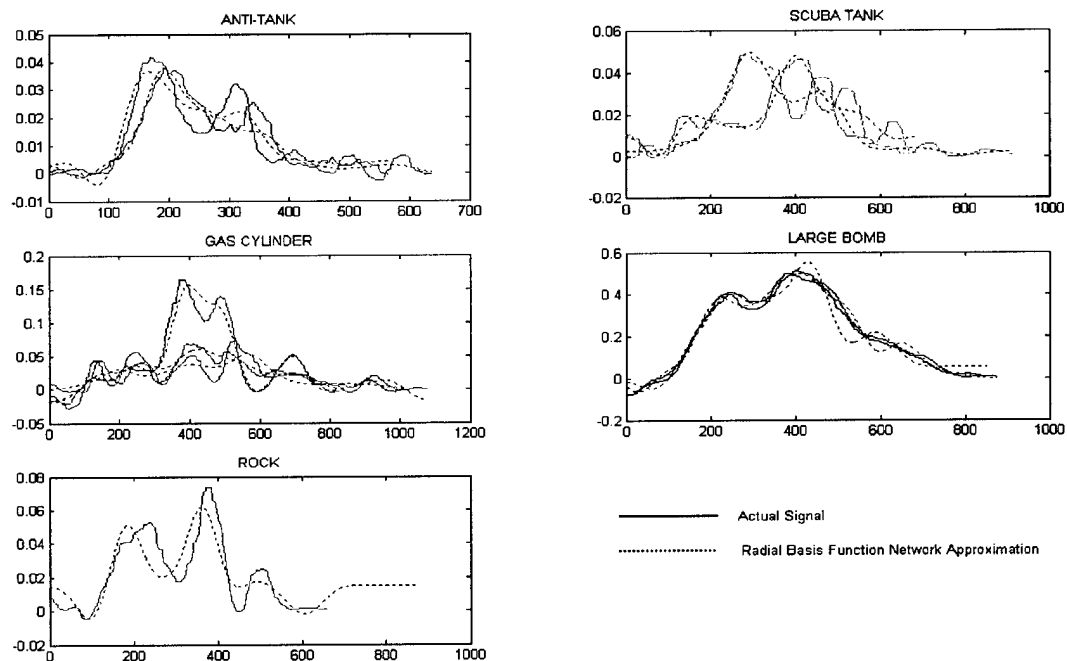


Figure 4.9. RBFN approximation of target signal using 9 hidden layers and a spread factor = 0.3.

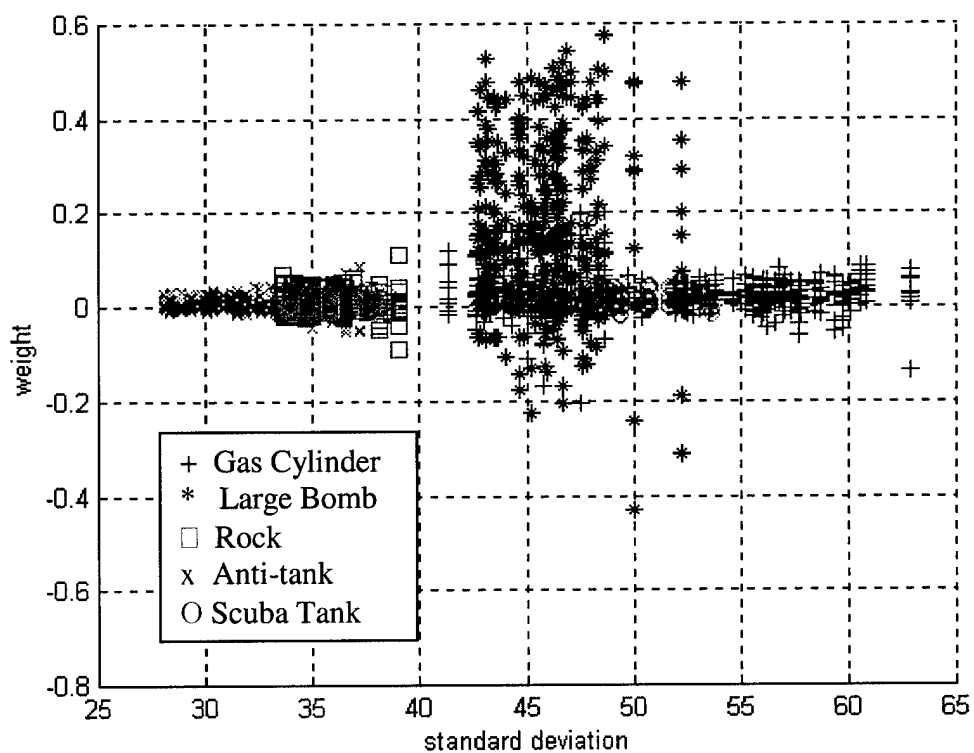


Figure 4.10. Weights vs widths of 20 trials per signal using an RBFN network generated by GMMs and the EM algorithm, 9 basis functions, spread scale = 0.3.

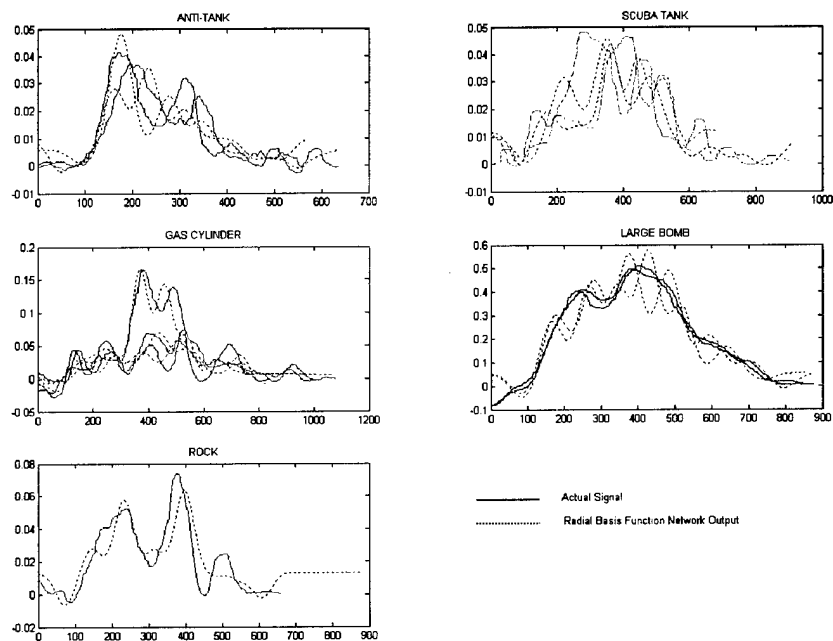


Figure 4.11. RBFN approximation of target signal using 8 basis functions and a spread factor of 0.12.

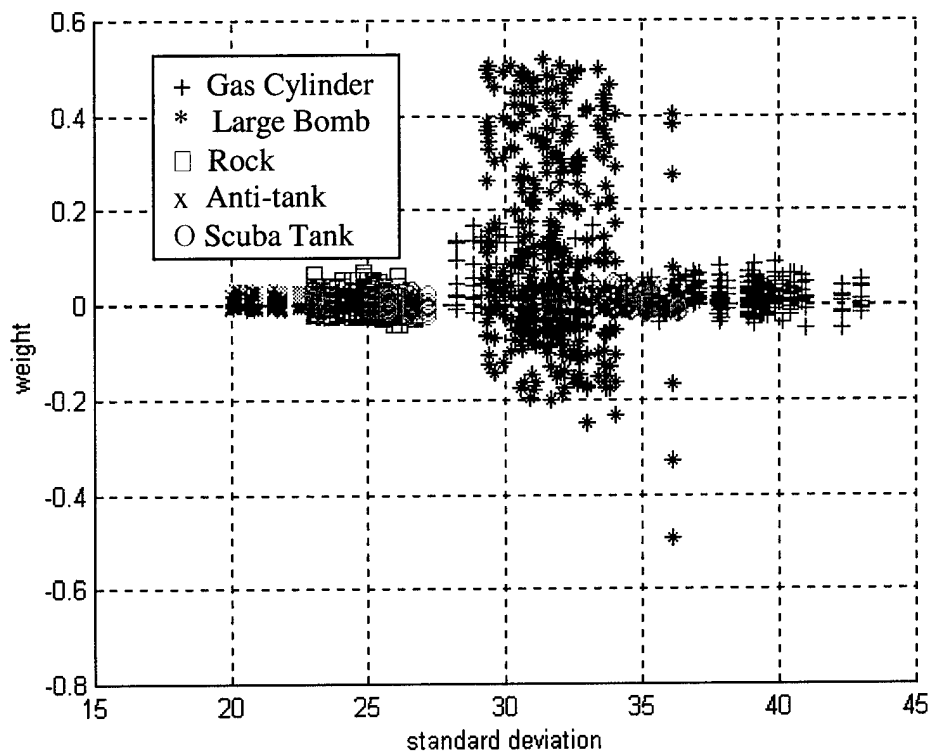


Figure 4.12. Weights vs widths of 20 trials per signal using an RBFN network generated by GMMs and the EM algorithm, 8 basis functions, spread scale = 0.12.

C. GAUSSIAN MIXTURE MODELS

When features have been identified and evaluated for clustering, the next step is to train a classifier using the extracted feature data. The ultimate goal is to use a trained classifier to statistically determine class membership with a high degree of confidence. A Gaussian Mixture Model (GMM) is one type of classifier that uses a weighted sum of Gaussian distributions to represent the feature vector distribution extracted from the target signal. The process was used in the NETLAB software to modeled signals as a sum of weighted Gaussians. It has been used extensively in classifying minelike targets using GPR data, speaker verification, and many other applications [Refs. 26, 27].

The GMM is defined as

$$p(\underline{x} | C_K) = \sum_{i=1}^M a_i p_i^{(K)}(\underline{x} | C_K), \quad (4.13)$$

where a_i is the mixture weight, M is the number of components and

$$p_i^{(K)}(\underline{x} | C_K) = N(\underline{m}_i^{(K)}, \Sigma_i^{(K)}). \quad (4.14)$$

The weights vector \underline{a}_i , mean vector $\underline{m}_i^{(k)}$, and the covariance matrix $\Sigma_i^{(k)}$ are unknown and have to be estimated. Sometimes the number of components, M , is known or can be assumed. The goal of the EM algorithm, which will be briefly explained in the next section, is to estimate these parameters that are not directly accessible to the user from the data. After the parameters are learned, unlabeled data can be compared against the GMM for each class and a class determination can be made.

1. Expectation-Maximization Algorithm

The pdf of the complete data set, \mathbf{y} , is specified as $p_y(\mathbf{y}; \theta)$, where θ is an unknown parameter vector. The samples \mathbf{y} cannot be directly observed though. What is observed are the samples $\mathbf{x} = \mathbf{g}(\mathbf{y})$, with a corresponding pdf $p_x(\mathbf{x}; \theta)$. This leads to the pdf of the incomplete data:

$$p_x(\mathbf{x};\theta) = \int_{Y(\mathbf{x})} p_y(\mathbf{y};\theta) d\mathbf{y}. \quad (4.15)$$

The maximum likelihood estimate of θ is given by

$$\hat{\theta}_{ML} : \sum_k \frac{\partial \ln(p_y(\mathbf{y}_k; \theta))}{\partial \theta} = 0. \quad (4.16)$$

However, all the \mathbf{y} 's are not available, so the EM algorithm maximizes the expectation of the log-likelihood function, conditioned on the observed samples and the current iteration estimate of θ . The two steps are:

- The Expectation step (E-step): At the $(t+1)$ th step of the iteration, where $\theta(t)$ is available, compute the expected value of

$$Q(\theta; \theta(t)) \equiv E \left[\sum_k \ln(p_y(\mathbf{y}_k; \theta | X; \theta(t))) \right]. \quad (4.17)$$

- The Maximization step (M-step): Compute the next $(t+1)$ th estimate of θ by maximizing $Q(\theta, \theta(t))$:

$$\theta(t+1) : \frac{\partial Q(\theta; \theta(t))}{\partial \theta} = 0. \quad (4.18)$$

An initial $\theta(t)$ is chosen and the E-step and M-step are performed successively until the algorithm converges. Convergence can be when parameters stop changing or when

$$\|\theta(t+1) - \theta(t)\| \leq \varepsilon \quad (4.19)$$

for some small ε and an appropriate distance measure. For a more detailed description of the EM algorithm, consult [Ref. 25].

Upon convergence of the EM algorithm, the features can be represented by a GMM, unique for each class. If enough trials had been available, the algorithm would have been applied to the feature data extracted from the target signals and would look similar to the visual representation as shown in Figure 4.13, where each cluster is represented a prior probability, mean (center of the cluster) and a covariance matrix representing the major and minor axis of the cluster.

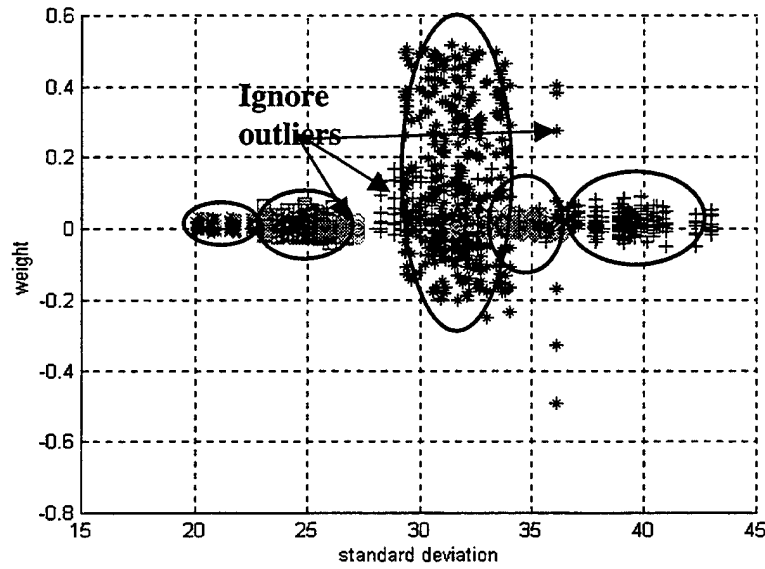


Figure 4.13. Hypothetical GMM used to represent features for 5 different classes.

2. Classification Step

After the classifier is trained, it can be used to identify unlabeled data. The decision rule for a classifier is a maximum-likelihood classifier which can be simplified down to

$$\hat{C} = \arg \max_{1 \leq c \leq C} \sum \log p(x_i | \lambda_c), \quad (4.20)$$

where $p(x_i | \lambda_c)$ is the gaussian distribution given in eq (4.14), specified by the parameters given λ_c for each class, x_i are the observed features from unlabeled data [Ref. 27]. A block diagram of a possible mine classification system using GMMs is given in Figure 4.14.

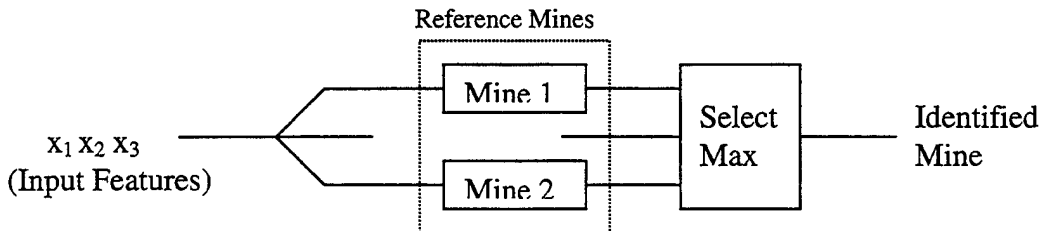


Figure 4.14. Mine Identification system. [Ref. 27]

The classifier compares the input feature vectors against the feature vectors for each class that has the maximum probability. The computations are straightforward and easily implemented because of the nature of the Gaussian distribution.

V. CONCLUSIONS AND RECOMMENDATIONS

A. CONCLUSIONS

This study presented a brief summary of the naval and land mine problem and some of the unique challenges they pose to naval forces trying to affect an expeditious and safe amphibious landing. One possible solution to the problem of mine clearance on the beach has been investigated by faculty in the physics department at the Naval Postgraduate School. The goal of this thesis to develop an automatic detection and classification algorithm that could run without the use of an expert operator. Many possible schemes were investigated, but the most simple and direct method proved to be the most successful. The short-term and zero-crossing rate detector algorithm developed correctly determined that a target signal was present and extracted the target portion of the signal for all man-made targets. Experiments conducted on a rock that is naturally found at the beach yielded only one signal identified as containing a target, showing potential for the apparatus used to acquire the signal and associated detection algorithms to be able to differentiate between man-made and natural features. The automatic detection algorithm needs to be tested on more data sets to gather quantitative data on its' success rate.

Various feature extraction methods were evaluated given the target signals extracted from the automatic detection algorithm. Three possible feature sets were identified out of a number of evaluated methods. The higher order moments, including the third, fourth, and fifth moment and the kurtosis and skewness show potential for interclass distinction. Modeling the target signals as the impulse response of a filter with poles and zeros determined by the Steiglitz-McBride iteration reconstructed the data signal and yielded zeros that might continue to show clustering between classes given more trials to evaluate. The third feature extraction method presented a GMM that represented the data as a sum of weighted Gaussian pulses and then used the weights and variances of the Gaussians as features.

B. RECOMMENDATIONS

A lot of signals that were collected during this experimental phase had to be discarded because they were collected using different seismic sources or recorded in a manner that was different from what was presented in this thesis. As a result more trials using the same experimental setup and targets need to be obtained to confirm the validity of the automatic detection algorithm and the evaluated feature extraction methods.

In addition, significant room for improvement and further experimentation exist. For example, it would be very useful to compare the response of a target upon initial burying and after several hours or up to a day of settling in the beach zone to determine the dependence on target burial time and a reasonably consistent target signal and features.

**APPENDIX A. MATLAB PROGRAMS (FOR FUNCTIONS
CALLED FROM NETLAB TOOLBOX CODE, SEE [REF. 24] FOR
CODE)**

APPENDIX A1. COMPUTATION OF THE COMPLEX POWER FROM RAW SIGNALS

```
% Name : rw_detect.m %
% July 11th 2000 by Jeremie Guy, intern %
% Modified 30 Aug 2000 LT Craig A Wilgenbusch
% using of the Vector Polarization Filtering methode in the detection
of Rayleigh %
% Waves %

clear all
cd s:\17NovData
for k=1:5

    load (strcat('cnt15',num2str(k),'a.mat'));
    load (strcat('cnt15',num2str(k),'b.mat'));
end

%Add signals from individual seismometers in array processing format
%Trial a
cnta_rad=cnt151a(:,1)+cnt152a(:,1)
        cnt153a(:,1)+cnt154a(:,1)+cnt155a(:,1);
cnta_ver=cnt151a(:,3)+cnt152a(:,3)
        cnt153a(:,3)+cnt154a(:,3)+cnt155a(:,3);

%Trial b
cntb_rad=cnt151b(:,1)+cnt152b(:,1)
        cnt153b(:,1)+cnt154b(:,1)+cnt155b(:,1);
cntb_ver=cnt151b(:,3)+cnt152b(:,3)
        cnt153b(:,3)+cnt154b(:,3)+cnt155b(:,3);

%Trial c
cntc_rad=cnt151c(:,1)+cnt152c(:,1)
        cnt153c(:,1)+cnt154c(:,1)+cnt155c(:,1);
cntc_ver=cnt151c(:,3)+cnt152c(:,3)
        cnt153c(:,3)+cnt154c(:,3)+cnt155c(:,3);

%Trial a
%Hilbert transform of the radial component of the signal%
radialh_a=hilbert(cnta_rad);
%Hilbert transform of the vertical component of the signal%
verticalh_a=hilbert(cnta_ver);
%Compute complex power
Cpower_nt_a=conj(radialh_a).*verticalh_a;

%Trial b
radialh_b=hilbert(cntb_rad);
verticalh_b=hilbert(cntb_ver);
Cpower_nt_b=conj(radialh_b).*verticalh_b;

%Trial c
radialh_c=hilbert(cntc_rad);
```

```

verticalh_c=hilbert(cntc_ver);
Cpower_nt_c=conj(radialh_c).*verticalh_c;

##### Data with Target #####

cd s:\13OctData
for k=1:5

    load (strcat('cwt15',num2str(k),'a.mat'));
    load (strcat('cwt15',num2str(k),'b.mat'));
    load (strcat('cwt15',num2str(k),'c.mat'));
end

%Add signals from individual seismometers in array processing format

%Trial a
cwta_rad=cwt151a(:,1)+cwt152a(:,1)+cwt153a(:,1)
+cwt154a(:,1)+cwt155a(:,1);
cwta_ver=cwt151a(:,3)+cwt152a(:,3)+cwt153a(:,3)
+cwt154a(:,3)+cwt155a(:,3);

%Trial b
cwtb_rad=cwt151b(:,1)+cwt152b(:,1)+cwt153b(:,1)
+cwt154b(:,1)+cwt155b(:,1);
cwtb_ver=cwt151b(:,3)+cwt152b(:,3)+cwt153b(:,3)
+cwt154b(:,3)+cwt155b(:,3);

%Trial c
cwtc_rad=cwt151c(:,1)+cwt152c(:,1)+cwt153c(:,1)
+cwt154c(:,1)+cwt155c(:,1);
cwtc_ver=cwt151c(:,3)+cwt152c(:,3)+cwt153c(:,3)
+cwt154c(:,3)+cwt155c(:,3);

%Coherence Subtraction, subtracting no target signal from target signal
cwta_rad_ch=cwta_rad-cnta_rad;
cwta_ver_ch=cwta_ver-cnta_ver;

cwtb_rad_ch=cwtb_rad-cntb_rad;
cwtb_ver_ch=cwtb_ver-cntb_ver;

cwtc_rad_ch=cwtc_rad-cntc_rad;
cwtc_ver_ch=cwtc_ver-cntc_ver;

%Hilbert transform of the radial component of the signal%
radialh_a=hilbert(cwta_rad);
%Hilbert transform of the vertical component of the signal%
verticalh_a=hilbert(cwta_ver);
Cpower_wt_a=conj(radialh_a).*verticalh_a;

radialh_b=hilbert(cwtb_rad);
verticalh_b=hilbert(cwtb_ver);
Cpower_wt_b=conj(radialh_b).*verticalh_b;

radialh_c=hilbert(cwtc_rad);
verticalh_c=hilbert(cwtc_ver);
Cpower_wt_c=conj(radialh_c).*verticalh_c;

```

```

%Chorence Subtraction complex power
%Hilbert transform of the radial component of the signal%
radialh_a=hilbert(cwta_rad_ch);
%Hilbert transform of the vertical component of the signal%
verticalh_a=hilbert(cwta_ver_ch);
Cpower_wt_a_ch=conj(radialh_a).*verticalh_a;

radialh_b=hilbert(cwtb_rad_ch);
verticalh_b=hilbert(cwtb_ver_ch);
Cpower_wt_b_ch=conj(radialh_b).*verticalh_b;

radialh_c=hilbert(cwtc_rad_ch);
verticalh_c=hilbert(cwtc_ver_ch);
Cpower_wt_c_ch=conj(radialh_c).*verticalh_c;

%Convert into real time index (seconds)
j=length(cwta_rad);
t=0:j/12800/(j-1):j/12800;

figure
subplot(3,1,1)
hold on
plot(t,imag(Cpower_nt_a),'r')
plot(t,imag(Cpower_nt_b),'b')
plot(t,imag(Cpower_nt_c),'k')
hold off
subplot(3,1,2)
hold on
plot(t,imag(Cpower_wt_a),'r')
plot(t,imag(Cpower_wt_b),'b')
plot(t,imag(Cpower_wt_c),'k')
hold off
subplot(3,1,3)
hold on
plot(t,imag(Cpower_wt_a_ch),'r')
plot(t,imag(Cpower_wt_b_ch),'b')
plot(t,imag(Cpower_wt_c_ch),'k')
hold off

Cpower_lb_a=Cpower_wt_a;
Cpower_lb_b=Cpower_wt_b;
Cpower_gc_c=Cpower_wt_c;
Cpower_gc_c_ch=Cpower_wt_c_ch;
Cpower_lb_b_ch1713=Cpower_wt_b_ch;
Cpower_lb_a_ch1713=Cpower_wt_a_ch;
save Cpower_lb_a_130Oct Cpower_lb_a
save Cpower_lb_b_130Oct Cpower_lb_b
save Cpower_gc_c_130Oct Cpower_gc_c
save Cpower_gc_c_ch_130Oct Cpower_gc_c_ch
save Cpower_lb_b_ch1713_130Oct Cpower_lb_b_ch1713
save Cpower_lb_a_ch1713_130Oct Cpower_lb_a_ch1713
save Cpower_nt_a_130Oct Cpower_nt_a
save Cpower_nt_b_130Oct Cpower_nt_b
save Cpower_nt_c_17Nov Cpower_nt_c

```

**APPENDIX A3. CODE USED TO DEMONSTRATE
COMPLEX POWER AND VECTOR POLARIZATION IN FIGURE
3.1.**

```
t=1:1:500;
r=cos(2*pi*t/50);
v(1:150)=cos(2*pi*t(1:150)/50);
v(151:300)=cos(2*pi*t(151:300)/50+pi/2);
v(301:500)=cos(2*pi*t(301:500)/50);
figure
subplot(3,1,1)
plot(r)
hold on
plot(v,'--r')
hold off
title('Time series')
xlabel('time')
ylabel('Amplitude')
legend('radial', 'vertical')
rh=hilbert(r);
vh=hilbert(v);
Cpower=conj(rh).*vh;
subplot(3,1,2)
plot(imag(Cpower));
title('Imaginary Complex Power')
xlabel('time')
ylabel('amplitude')
subplot(3,1,3)
plot(real(Cpower));
title('Real Complex Power')
xlabel('time')
ylabel('amplitude')
```

APPENDIX A4. SHORT-TERM ENERGY AND ZERO-CROSSING RATE AUTOMATIC DETECTOR

```
%Filename: ss_detect.m
%Written by: C. Wilgenbusch
%Date last modified: 10 Dec 2000
%Purpose: Uses a short term energy and a zero-crossing rate detector
%to detect a possible target signal from the imaginary complex power
%signal. If the target is present, it extracts the portion of the
%signal believed to be a target.

clear all;
load e:\ComplexPowerData\Cpower_gc_a_ch_17Nov
x=Cpower_gc_a_ch;
fs=12800;

[ste,t1]=sp_steng(imag(x),25/12800,50,fs);
[stzc,t2]=sp_stzcr(imag(x),25/12800,50,fs);

ste=ste/max(ste);
stzc=stzc/max(stzc);

figure
subplot(2,1,1)
plot(t1,ste)
title('Short term energy of signal')
subplot(2,1,2)
plot(t2,stzc)
title('Short term zero-crossing rate')

start=75;
signal_found=0;
signal_end=0;

while signal_found == 0 & (start+25<length(stzc))
    zero_avg=sum(stzc(start:start+25));
    if (zero_avg == 0 & ste(start) > 0.01)
        signal_found=1;
    else
        start=start+1;
    end
end

if signal_found == 0
    disp('NO TARGET FOUND')
else
    ending=start;
    start=start-10;
    while signal_end == 0
```

```

    zero_avg=sum(stzc(ending:ending+20));
    if ste(ending) < 0.0001 & zero_avg > 0
        signal_end=1;
    else
        ending=ending+1;
    end
end

%Translate t1 into coherence time.

j=length(x);
t_ch=0:j/12800:(j-1):j/12800;

index_start=t1(start)*fs;
index_end=t1(ending)*fs;

st_target_b=-imag(x(index_start:index_end));
save s:\st_target_b st_target_b
subplot(2,1,1)
plot(t1(start:ending),ste(start:ending));
title('Extracted STE portion of signal')
subplot(2,1,2)
plot(t_ch(index_start:index_end),imag(x(index_start:index_end)));
title('Extracted Complex Power of signal')
end

```


APPENDIX A5. SHORT-TERM ENERGY ROUTINE, CODE WRITTEN BY D. BROWN

```

function [y,tscale] = sp_steng(x,frame,overlap,fs>window)
%SP_STENG Short-time energy.
%
%   [Y,TSCALE] = SP_STENG(X,FRAME,OVERLAP,FS) computes
%   the short-time energy of X using a size FRAME (mili-
%   seconds) and a percentage OVERLAP between successive
%   frames using a rectangular data window. The sampling
%   frequency is given by FS. The short-time energy
%   curve is returned in Y and a time scale corresponding
%   to the end of the data frame is returned in TSCALE.
%   The curve may be displayed with the command
%   'plot(y,tscale)'.
%
%   [Y,TSCALE] = SP_STENG(X,FRAME,OVERLAP,FS,'WINDOW')
%   windows the data through the specified 'WINDOW' before
%   computing the short-time energy. 'WINDOW' can be
%   one of the following: 'hamming', 'hanning', 'bartlett',
%   'blackman' or 'triang'.
%
%   See also: SP_STENG, SP_STZCR, AVSMOOTH, MDSMOOTH
%
%   SP_STENG is implemented as a mex function on some
%   installations.
%
%   LT Dennis W. Brown 7-11-93, DWB 11-11-94
%   Naval Postgraduate School, Monterey, CA
%   May be freely distributed.
%   Not for use in commercial products.
%
%   Ref: Rabiner & Schafer, Digital Processing of Speech
%   Signals, 1978, ss 4.2, pp 120-126.
%
% default values
y = [];tscale=[];

% must have at least 3 args
if nargin < 4
    error('sp_steng: Requires first three arguments.');
```

```

end;

% percentage must be in range 0-100
if overlap < 0 | overlap > 100,
    error('sp_steng: Overlap percentage must be in range 0-100%');
end;

% figure out if we have a vector
if min(size(x)) ~= 1,
    error('sp_steng: Input arg "x" must be a 1xN or Nx1 vector.');
```

```

end;

```

```

% work with Nx1 vectors
x = x(:);

% variables
Ns = length(x); % number of samples
N = floor(fs * frame); % samples-per-frame
Ndiff = floor(N * (1 - overlap/100)); % samples between windows
L = floor((Ns-N)/Ndiff); % number of windows
y = zeros(L,1); % space for answer
tscale = zeros(L,1); % space for indices

% data window
datawindow = ones(N,1); % rectangular default
if nargin == 5
    if strcmp(window,'hamming')
        datawindow = hamming(N);
    elseif strcmp(window,'hanning')
        datawindow = hanning(N);
    elseif strcmp(window,'blackman')
        datawindow = blackman(N);
    elseif strcmp(window,'bartlett')
        datawindow = bartlett(N);
    elseif strcmp(window,'triang')
        datawindow = triang(N);
    end;
end;

% square the data and the window
datawindow = datawindow .^ 2;
x = x .^ 2;

% windows with overlap
for k=1:L
    s1 = (k-1) * Ndiff + 1; % start of window
    tscale(k,1) = k * Ndiff/fs;
    y(k,1) = sum(x(s1:s1+N-1,1) .* datawindow);
end;

```

APPENDIX A6. ZERO-CROSSING RATE ROUTINE, CODE WRITTEN BY D. BROWN

```

function [y,tscale] = sp_stzcr(x,frame,overlap,fs>window)
%SP_STZCR Short-time zero crossings.
%   [Y,TSCALE] = SP_STZCR(X,FRAME,OVERLAP,FS) computes the
%   short-time zero-crossing rate of X using a frame size
%   of LENGTH and a percentage OVERLAP between successive
%   frames using a rectangular data window. The sampling
%   frequency is given by FS. The short-time zero-crossing
%   curve is returned in Y and a time scale corresponding
%   to the end of the data frame is returned in TSCALE.
%   The curve may be displayed with the command
%   'plot(y,tscale)'.
%
%   See also: SP_STMAG, SP_STZCR, AVSMOOTH, MDSMOOTH
%
%   SP_STZCR is implemented as a mex function on some
%   installations.
%
%   LT Dennis W. Brown 7-11-93, DWB 11-11-94
%   Naval Postgraduate School, Monterey, CA
%   May be freely distributed.
%   Not for use in commercial products.
%
%   Ref: Rabiner & Schafer, Digital Processing of Speech
%   Signals, 1978, ss 4.3, pp 127-130.
%
% window argument is not used but is here to maintain consistency with
% the other sp_ routines.
%
% default values
y = [];tscale = [];
%
% must have at least 3 args
if nargin < 4
    error('sp_stzcr: Requires first four arguments.');
```

```

end
%
% figure out if we have a vector
if min(size(x)) ~= 1,
    error('sp_stzcr: Input arg "x" must be a 1xN or Nx1 vector.');
```

```

end;
%
% work with Nx1 vectors
x = x(:);
%
% variables
Ns = length(x);
N = floor(fs * frame);
Ndiff = floor(N * (1 - overlap/100));
```

```

% number of samples
% samples per frame
% samples between windows
```

```

L = floor((Ns-N)/Ndiff);           % number of windows
y = zeros(L,1);                   % space for answer
tscale = zeros(L,1);              % space for time

% use the absolute value of x
t = abs( sign( x(2:Ns,1) ) - sign( x(1:Ns-1,1) ) );

% windows with overlap
for k=1:L
    s1 = (k-1) * Ndiff + 1;        % start of window
    tscale(k,1) = k * Ndiff/fs;
    y(k,1) = sum(t(s1:s1+N-1,1))/2/N;
end

```

APPENDIX A7. EVALUATION OF HIGHER ORDER MOMENTS AS POTENTIAL FEATURES

```
%Filename: ss_detect.m
%Written by: C. Wilgenbusch
%Date last modified: 01 MAR 2001
%Purpose: Calculate the 3rd, 4th, 5th moments, kurtosis and skewness
%of the data and plot for evaluation as a possible feature.
```

```
clear all
```

```
%Load the target data extracted from ss_detect.m
load at_target_a
load at_target_b
load st_target_a
load st_target_b
load gc_target_a
load gc_target_b
load gc_target_c
load lb_target_a
load lb_target_b
load rc_target_c
```

```
%Limit signal to shortest signal
at_a=at_target_a(1:565);
at_b=at_target_b(1:565);
st_a=st_target_a(1:565);
st_b=st_target_b(1:565);
gc_a=gc_target_a(1:565);
gc_b=gc_target_b(1:565);
gc_c=gc_target_c(1:565);
lb_a=lb_target_a(1:565);
lb_b=lb_target_b(1:565);
rc_c=rc_target_c(1:565);
```

```
%Remove the mean
at_a=at_a-mean(at_a);
at_b=at_b-mean(at_b);
st_a=st_a-mean(st_a);
st_b=st_b-mean(st_b);
gc_a=gc_a-mean(gc_a);
gc_b=gc_b-mean(gc_b);
gc_c=gc_c-mean(gc_c);
lb_a=lb_a-mean(lb_a);
lb_b=lb_b-mean(lb_b);
rc_c=rc_c-mean(rc_c);
```

```
%Normalize to Unit Energy
at_a=at_a/(norm(at_a,2));
at_b=at_b/(norm(at_b,2));
gc_a=gc_a/(norm(gc_a,2));
```

```

gc_b=gc_b/(norm(gc_b,2));
st_a=st_a/(norm(st_a,2));
st_b=st_b/(norm(st_b,2));
gc_c=gc_c/(norm(gc_c,2));
lb_a=lb_a/(norm(lb_a,2));
lb_b=lb_b/(norm(lb_b,2));
rc_c=rc_c/(norm(rc_c,2));

%Calculate higher order moments
[at_am2,at_am3,at_am4,at_am5]=mm(at_a);
[at_bm2,at_bm3,at_bm4,at_bm5]=mm(at_b);
[st_am2,st_am3,st_am4,st_am5]=mm(st_a);
[st_bm2,st_bm3,st_bm4,st_bm5]=mm(st_b);
[gc_cm2,gc_cm3,gc_cm4,gc_cm5]=mm(gc_c);
[gc_am2,gc_am3,gc_am4,gc_am5]=mm(gc_a);
[gc_bm2,gc_bm3,gc_bm4,gc_bm5]=mm(gc_b);
[lb_am2,lb_am3,lb_am4,lb_am5]=mm(lb_a);
[lb_bm2,lb_bm3,lb_bm4,lb_bm5]=mm(lb_b);
[rc_cm2,rc_cm3,rc_cm4,rc_cm5]=mm(rc_c);

%Put results in Matrix form
MM=zeros(10,6);
MM(1,:)=[at_am2,at_am3,at_am4,at_am5,kurtosis(at_a),skewness(at_a)];
MM(2,:)=[at_bm2,at_bm3,at_bm4,at_bm5,kurtosis(at_b),skewness(at_b)];
MM(3,:)=[st_am2,st_am3,st_am4,st_am5,kurtosis(st_a),skewness(st_a)];
MM(4,:)=[st_bm2,st_bm3,st_bm4,st_bm5,kurtosis(st_b),skewness(st_b)];
MM(5,:)=[gc_am2,gc_am3,gc_am4,gc_am5,kurtosis(gc_a),skewness(gc_a)];
MM(6,:)=[gc_bm2,gc_bm3,gc_bm4,gc_bm5,kurtosis(gc_b),skewness(gc_b)];
MM(7,:)=[gc_cm2,gc_cm3,gc_cm4,gc_cm5,kurtosis(gc_c),skewness(gc_c)];
MM(8,:)=[lb_am2,lb_am3,lb_am4,lb_am5,kurtosis(lb_a),skewness(lb_a)];
MM(9,:)=[lb_bm2,lb_bm3,lb_bm4,lb_bm5,kurtosis(lb_b),skewness(lb_b)];
MM(10,:)=[rc_cm2,rc_cm3,rc_cm4,rc_cm5,kurtosis(rc_c),skewness(rc_c)];

%Plot results
figure
hold on
plot3(MM(1,2),MM(1,4),MM(1,6),'db')
plot3(MM(2,2),MM(2,4),MM(2,6),'db')
plot3(MM(3,2),MM(3,4),MM(3,6),'+r')
plot3(MM(4,2),MM(4,4),MM(4,6),'+r')
plot3(MM(5,2),MM(5,4),MM(5,6),'*g')
plot3(MM(6,2),MM(6,4),MM(6,6),'*g')
plot3(MM(7,2),MM(7,4),MM(7,6),'*g')
plot3(MM(8,2),MM(8,4),MM(8,6),' .k')
plot3(MM(9,2),MM(9,4),MM(9,6),' .k')
plot3(MM(10,2),MM(10,4),MM(10,6),'om')
hold off
title('Third Moment vs Fifth Moment vs Skewness, signals normalized to unit energy');
xlabel('Third Moment')
ylabel('Fifth Moment')
zlabel('Skewness')
legend('Gas Cylinder',' ',' ','Anti-tank',' ','Scuba Tank',' ','Large Bomb',' ','Rock')
grid on

```

APPENDIX A8. CODE USED TO CALCULATE MOMENTS

```
function [m2,m3,m4,m5]=mm(x)
m2=var(x);
m3=sum(x.^3)/l;
m4=sum(x.^4)/l;
m5=sum(x.^5)/l;
```

APPENDIX A9. CODE USED TO CALCULATE MOMENTS

```
% Filename: wav03.m
% Author: Monique P. Fargues
% Written: 1/26/01
% Purpose: Model target data as impulse response of filter
% using stmcb.m and plot poles and zeros to evaluate for use
% as features.

iopt=2;
clear ERR RECON P1 data
ERR=[];RECON=[];P1=[];RECONC=[];ERRC=[];
if idec==1,npt=512;else,npt=1024;end
data(:,:,1)=gc_target; % data(:,ktrial,ksig)
data(:,:,2)=[at_target,zeros(npt,1)];
data(:,:,3)=[st_target,zeros(npt,1)];
data(:,:,4)=[lb_target,zeros(npt,1)];
data(:,:,5)=[rc_target,zeros(npt,2)];
ntrial=[3,2,2,2,1];
l_dat % length of data (ksig,,ktrial)
clear A AR B BR ar br
RECON=zeros(1024,3,5);
for ksig=1:5
    ksig
    for ktrial=1:ntrial(ksig)
        f=data(1:l_dat(ksig,ktrial),ktrial,ksig);
        [b,a]=stmcb(f,6,4,8);
        recon=filter(b,a,[1;zeros(l_dat(ksig,ktrial)-1,1)]);
        err=norm(recon-f);
        ERR(ktrial,ksig)=err/(sqrt(norm(recon)*norm(f)));
        A(:,ktrial,ksig)=a; AR(:,ktrial,ksig)=roots(a);
        RECON(1:l_dat(ksig,ktrial),ktrial,ksig)=recon;
        B(:,ktrial,ksig)=b;BR(:,ktrial,ksig)=roots(b);
        plot([f,recon])
        pause
    end
end
end
ii=0;

% plot poles/zeros
% BR(:,ktrial,ksig)=roots(b);
kcount=1;figure
```


APPENDIX A10. CODE USED TO LOAD AND PROCESS INPUT DATA

```
% Filename: indata.m
% Author: Monique P. Fargues
% Written: 1/26/01
% Purpose: load target data and prepare for use in stmcb.m

len_mds=12;iarg=0;idec=0;
load gc_target_a.mat;ngca=length(gc_target_a);
load gc_target_b.mat;ngcb=length(gc_target_b);
load
gc_target_c.mat;gc_target_c=gc_target_c(1:1024);ngcc=length(gc_target_c
);l_gc=[ngca,ngcb,ngcc];
length_gc=min(min(length(gc_target_a),length(gc_target_b)),length(gc_ta
rget_c));
gc_target_a=process_sig(gc_target_a,len_mds,idec,iarg);
gc_target_b=process_sig(gc_target_b,len_mds,idec,iarg);
gc_target_c=process_sig(gc_target_c,len_mds,idec,iarg);
gc_target=[gc_target_a,gc_target_b,gc_target_c];

load at_target_a.mat
load at_target_b.mat;l_at=[length(at_target_a),length(at_target_b)];

%load at_target_c.mat
length_at=min(length(at_target_a),length(at_target_b));
at_target_a=process_sig(at_target_a,len_mds,idec,iarg);
at_target_b=process_sig(at_target_b,len_mds,idec,iarg);
at_target=[at_target_a,at_target_b];

load st_target_a.mat
load st_target_b.mat
%load st_target_c.mat
length_st=min(length(st_target_a),length(st_target_b));
l_st=[length(st_target_a),length(st_target_b)];
st_target_a=process_sig(st_target_a,len_mds,idec,iarg);
st_target_b=process_sig(st_target_b,len_mds,idec,iarg);
%shift scuba tank signal a
temp=zeros(1,1024);
temp(1:984)=st_target_a(41:1024);
temp(985:end)=st_target_a(1:40);
st_target_a=temp';
st_target=[st_target_a,st_target_b];

load lb_target_a.mat
load lb_target_b.mat
%load lb_target_c.mat
l_lb=[length(lb_target_a),length(lb_target_b)];
length_lb=min(length(lb_target_a),length(lb_target_b));
lb_target_a=process_sig(lb_target_a,len_mds,idec,iarg);
lb_target_b=process_sig(lb_target_b,len_mds,idec,iarg);
```

```

lb_target=[lb_target_a,lb_target_b];

%load rc_target_a.mat
%load rc_target_b.mat
load rc_target_c.mat
l_rc=length(rc_target_c);
rc_target=process_sig(rc_target_c,len_mds,idec,iarg);
l_dat=[l_gc;l_at,0;l_st,0;l_lb,0;l_rc,0,0];
if idec==1,l_dat=fix(l_dat/2.);end

subplot(321),plot(gc_target_a),hold on
plot(gc_target_b),plot(gc_target_c)
hold off

subplot(322),plot(at_target_a),hold on
plot(at_target_b)
hold off
subplot(323),plot(st_target_a),hold on
plot(st_target_b),hold off

subplot(324),plot(lb_target_a),hold on
plot(lb_target_b),hold off

subplot(325),plot(rc_target_c)

% Filename: process_sig.m
% Author: Monique P. Fargues
% Written: 1/26/01
% Purpose: smooth data and normalize to unit energy

function y=process_sig(x,len_mds,idec,iarg)

l_x=length(x);
if iarg==0,y0=[mdsmooth(x,len_mds);zeros(1024-l_x,1)];
elseif iarg==1,y0=[mdsmooth(x,len_mds);zeros(1024-l_x,1)];end
y=y0;
if idec==1,y=decimate(y0,2);end
y=y-mean(y);y=y/(norm(y,2)); % normalize signal to unit energy
return

```

APPENDIX A11. RADIAL BASIS FUNCTION MODELING CODE

```
% Filename: test_rad2.m
% Author: LT Craig A. Wilgenbusch
% Written: 3/02/01
% Purpose: Model target data as a mixture of Gaussian
% shaped pulses using routines from NETLAB software
% toolbox and plot features (widths and weights)
% for evaluations as features

clear all

load at_target_a
load at_target_b.mat

load st_target_a.mat
load st_target_b.mat

load gc_target_a.mat
load gc_target_b.mat
load gc_target_c.mat

load lb_target_a.mat
load lb_target_b.mat

load rc_target_c.mat

at_target_a=mdsmooth(at_target_a,25);
at_target_b=mdsmooth(at_target_b,25);

st_target_a=mdsmooth(st_target_a,45);
st_target_b=mdsmooth(st_target_b,60);

gc_target_a=mdsmooth(gc_target_a,25);
gc_target_b=mdsmooth(gc_target_b,25);
gc_target_c=mdsmooth(gc_target_c,25);

lb_target_a=mdsmooth(lb_target_a,25);
lb_target_b=mdsmooth(lb_target_b,25);

rc_target_c=mdsmooth(rc_target_c,25);

P_at_a=(1:length(at_target_a))';
P_at_b=(1:length(at_target_b))';

P_st_a=(1:length(st_target_a))';
P_st_b=(1:length(st_target_b))';

P_gc_a=(1:length(gc_target_a))';
P_gc_b=(1:length(gc_target_b))';
P_gc_c=(1:length(gc_target_c))';
```

```

P_lb_a=(1:length(lb_target_a))';
P_lb_b=(1:length(lb_target_b))';

P_rc_c=(1:length(rc_target_c))';

% Set up network parameters.
nin = 1;          % Number of inputs.
nhidden = 9;      % Number of hidden units.
nout = 1;         % Number of outputs.

options = foptions;
options(1) = 0;    % Display EM training
options(14) = 5;   % number of iterations of EM
options(7)=.3;
trials=1;
cent=zeros(nhidden,10,trials);
wi=zeros(nhidden,10,trials);
w2=zeros(nhidden,10,trials);

for j=1:trials
    j
    %Anti-Tank
    % Create and initialize network weight and parameter vectors.
    net = rbf(nin, nhidden, nout, 'gaussian');
    net = rbftrain(net, options, P_at_a, at_target_a);
    y = rbffwd(net, P_at_a);

    net2 = rbf(nin, nhidden, nout, 'gaussian');
    net2 = rbftrain(net2, options, P_at_b, at_target_b);
    y2 = rbffwd(net2, P_at_b);

    if j == 1
        figure
        subplot(3,2,1)
        plot(at_target_a,'r')
        hold on
        plot(y,':')

        title('ANTI-TANK')
        plot(at_target_b,'r')
        plot(y2,':')
        hold off
    end

    cent(:,1,j)=net.c;
    cent(:,2,j)=net2.c;
    wi(:,1,j)=net.wi';
    wi(:,2,j)=net2.wi';
    w2(:,1,j)=net.w2;
    w2(:,2,j)=net2.w2;

    %SCUBA TANK
    % Create and initialize network weight and parameter vectors.
    net = rbf(nin, nhidden, nout, 'gaussian');
    net = rbftrain(net, options, P_st_a, st_target_a);

```

```

y = rbffwd(net, P_st_a);

net2 = rbf(nin, nhidden, nout, 'gaussian');
net2 = rbftrain(net2, options, P_st_b, st_target_b);
y2 = rbffwd(net2, P_st_b);

if j == 1

subplot(3,2,2)
plot(st_target_a,'g')
hold on
plot(y,':')
title('SCUBA TANK')
plot(st_target_b,'g')
plot(y2,':')
hold off
end

cent(:,3,j)=net.c;
cent(:,4,j)=net2.c;
wi(:,3,j)=net.wi';
wi(:,4,j)=net2.wi';
w2(:,3,j)=net.w2;
w2(:,4,j)=net2.w2;

clear net net2

clear net net2

%Gas Cylinder Tank
% Create and initialize network weight and parameter vectors.
net = rbf(nin, nhidden, nout, 'gaussian');
net = rbftrain(net, options, P_gc_a, gc_target_a);
y = rbffwd(net, P_gc_a);

net2 = rbf(nin, nhidden, nout, 'gaussian');
net2 = rbftrain(net2, options, P_gc_b, gc_target_b);
y2 = rbffwd(net2, P_gc_b);

net3 = rbf(nin, nhidden, nout, 'gaussian');
net3 = rbftrain(net3, options, P_gc_c, gc_target_c);
y3 = rbffwd(net3, P_gc_c);

if j ==1

subplot(3,2,3)
plot(gc_target_a)
hold on
plot(y,':')

title('GAS CYLINDER')

plot(gc_target_b)

```

```

plot(y2,':')
plot(gc_target_c)
plot(y3,':')
hold off
end

cent(:,5,j)=net.c;
cent(:,6,j)=net2.c;
cent(:,7,j)=net3.c;
wi(:,5,j)=net.wi';
wi(:,6,j)=net2.wi';
wi(:,7,j)=net3.wi';
w2(:,5,j)=net.w2;
w2(:,6,j)=net2.w2;
w2(:,7,j)=net3.w2;
b(5)=net.b2;

clear net net2 net3

%LARGE BOMB
% Create and initialize network weight and parameter vectors.
net = rbf(nin, nhidden, nout, 'gaussian');
net = rbftrain(net, options, P_lb_a, lb_target_a);
y = rbffwd(net, P_lb_a);

net2 = rbf(nin, nhidden, nout, 'gaussian');
net2 = rbftrain(net2, options, P_lb_b, lb_target_b);
y2 = rbffwd(net2, P_lb_b);

if j==1

subplot(3,2,4)
plot(lb_target_a,'k')
hold on
plot(y,':')

title('LARGE BOMB')

plot(lb_target_b,'k')
plot(y2,':')
hold off
end

cent(:,8,j)=net.c;
cent(:,9,j)=net2.c;
wi(:,8,j)=net.wi';
wi(:,9,j)=net2.wi';
w2(:,8,j)=net.w2;
w2(:,9,j)=net2.w2;

clear net net2

%ROCK
% Create and initialize network weight and parameter vectors.

```

```

net = rbf(nin, nhhidden, nout, 'gaussian');
net = rbftrain(net, options, P_rc_c, rc_target_c);
y = rbffwd(net, P_lb_a);

if j==1
subplot(3,2,5)
plot(rc_target_c,'m')
hold on
plot(y,':')
hold off
title('ROCK')
end

cent(:,10,j)=net.c;
wi(:,10,j)=net.wi';
w2(:,10,j)=net.w2;

clear net

end

figure

hold on
for j=1:trials

    plot(sqrt(wi(:,1,j)),w2(:,1,j),'xr');
    plot(sqrt(wi(:,2,j)),w2(:,2,j),'xr');
    plot(sqrt(wi(:,3,j)),w2(:,3,j),'og');
    plot(sqrt(wi(:,4,j)),w2(:,4,j),'og');
    plot(sqrt(wi(:,5,j)),w2(:,5,j),'+');
    plot(sqrt(wi(:,6,j)),w2(:,6,j),'+');
    plot(sqrt(wi(:,7,j)),w2(:,7,j),'+');
    plot(sqrt(wi(:,8,j)),w2(:,8,j),'*k');
    plot(sqrt(wi(:,9,j)),w2(:,9,j),'*k');
    plot(sqrt(wi(:,10,j)),w2(:,10,j),'sm');

end
hold off
grid on
clear cent wi w2

```

APPENDIX A12. NETLAB TOOLBOX [REF 24] CODE *rbf.m* USED TO CREATE AND INITIALIZE NETWORK

```
function net = rbf(nin, nhidden, nout, rbfunc, outfunc, prior, beta)
%RBF Creates an RBF network with specified architecture
%
% Description
% NET = RBF(NIN, NHIDDEN, NOUT, RBFUNC) constructs and initialises a
% radial basis function network returning a data structure NET. The
% weights are all initialised with a zero mean, unit variance normal
% distribution, with the exception of the variances, which are set to
% one. This makes use of the Matlab function RANDN and so the seed for
% the random weight initialization can be set using RANDN('STATE', S)
% where S is the seed value. The activation functions are defined in
% terms of the distance between the data point and the corresponding
% centre. Note that the functions are computed to a convenient
% constant multiple: for example, the Gaussian is not normalised.
% (Normalisation is not needed as the function outputs are linearly
% combined in the next layer.)
%
% The fields in NET are
%   type = 'rbf'
%   nin = number of inputs
%   nhidden = number of hidden units
%   nout = number of outputs
%   nwts = total number of weights and biases
%   actfn = string defining hidden unit activation function:
%   'gaussian' for a radially symmetric Gaussian function.
%   'tps' for  $r^2 \log r$ , the thin plate spline function.
%   'r4logr' for  $r^4 \log r$ .
%   outfn = string defining output error function:
%   'linear' for linear outputs (default) and SoS error.
%   'neuroscale' for Sammon stress measure.
%   c = centres
%   w1 = squared widths (null for rlogr and tps)
%   w2 = second layer weight matrix
%   b2 = second layer bias vector
%
% NET = RBF(NIN, NHIDDEN, NOUT, RBFUNC, OUTFUNC) allows the user to
% specify the type of error function to be used. The field OUTFN is
% set to the value of this string. Linear outputs (for regression
% problems) and Neuroscale outputs (for topographic mappings) are
% supported.
%
% NET = RBF(NIN, NHIDDEN, NOUT, RBFUNC, OUTFUNC, PRIOR, BETA), in
% which
% PRIOR is a scalar, allows the field NET.ALPHA in the data structure
% NET to be set, corresponding to a zero-mean isotropic Gaussian prior
% with inverse variance with value PRIOR. Alternatively, PRIOR can
% consist of a data structure with fields ALPHA and INDEX, allowing
% individual Gaussian priors to be set over groups of weights in the
% network. Here ALPHA is a column vector in which each element
% corresponds to a separate group of weights, which need not be
% mutually exclusive. The membership of the groups is defined by the
% matrix INDX in which the columns correspond to the elements of
% ALPHA.
```



```

% Each column has one element for each weight in the matrix, in the
% order defined by the function MLPPAK, and each element is 1 or 0
% according to whether the weight is a member of the corresponding
% group or not. A utility function RBFPRIOR is provided to help in
% setting up the PRIOR data structure.
%
% NET = RBF(NIN, NHIDDEN, NOUT, FUNC, PRIOR, BETA) also sets the
% additional field NET.BETA in the data structure NET, where beta
% corresponds to the inverse noise variance.
%
% See also
% RBFERR, RBFFWD, RBFGRAD, RBFPK, RBFTRAIN, RBFUNPAK
%

```

```

% Copyright (c) Ian T Nabney (1996-9)

```

```

net.type = 'rbf';
net.nin = nin;
net.nhidden = nhidden;
net.nout = nout;

```

```

% Check that function is an allowed type
actfns = {'gaussian', 'tps', 'r4logr'};
outfns = {'linear', 'neuroscale'};
if (strcmp(rbfnc, actfns)) == 0
    error('Undefined activation function.')
else
    net.actfn = rbfnc;
end
if nargin <= 4
    net.outfn = outfns{1};
elseif (strcmp(outfnc, outfns) == 0)
    error('Undefined output function.')
else
    net.outfn = outfnc;
end

```

```

% Assume each function has a centre and a single width parameter, and
that
% hidden layer to output weights include a bias. Only the Gaussian
function

```

```

% requires a width
net.nwts = nin*nhidden + (nhidden + 1)*nout;
if strcmp(rbfnc, 'gaussian')
    % Extra weights for width parameters
    net.nwts = net.nwts + nhidden;
end

```

```

if strcmp(net.outfn, 'neuroscale')
    net.mask = rbfprior(rbfnc, nin, nhidden, nout);
end

```

```

if nargin > 5
    if isstruct(prior)
        net.alpha = prior.alpha;
        net.index = prior.index;
    elseif size(prior) == [1 1]

```

```

    net.alpha = prior;
else
    error('prior must be a scalar or a structure');
end
end

w = randn(1, net.nwts);
outfunc = net.outfn;
net.outfn = 'linear';
net = rbfunpak(net, w);
net.outfn = outfunc;

% Make widths equal to one
if strcmp(rbfunc, 'gaussian')
    net.wi = ones(1, nhidden);
end

```

APPENDIX A13. NETLAB TOOLBOX [REF 24] CODE *rbftrain.m* USED TO TRAIN NETWORK USING THE EM ALGORITHM AND A GMM MODEL.

```

function [net, options] = rbftrain(net, options, x, t)
%RBFTRAIN Two stage training of RBF network.
%
% Description
% NET = RBFTRAIN(NET, OPTIONS, X, T) uses a two stage training
% algorithm to set the weights in the RBF model structure NET. Each
row
% of X corresponds to one input vector and each row of T contains the
% corresponding target vector. The centres are determined by fitting a
% Gaussian mixture model with circular covariances using the EM
% algorithm through a call to RBFSETBF. (The mixture model is
% initialised using a small number of iterations of the K-means
% algorithm.) If the activation functions are Gaussians, then the
basis
% function widths are then set to the maximum inter-centre squared
% distance.
%
% For linear outputs, the hidden to output weights that give rise to
% the least squares solution can then be determined using the pseudo-
% inverse. For neuroscale outputs, the hidden to output weights are
% determined using the iterative shadow targets algorithm. Although
% this two stage procedure may not give solutions with as low an error
% as using general purpose non-linear optimisers, it is much faster.
%
% The options vector may have two rows: if this is the case, then the
% second row is passed to RBFSETBF, which allows the user to specify a
% different number iterations for RBF and GMM training. The optional
% parameters to RBFTRAIN have the following interpretations.
%
% OPTIONS(1) is set to 1 to display error values during EM training.
%
% OPTIONS(2) is a measure of the precision required for the value of
% the weights W at the solution.

```

```

%
% OPTIONS(3) is a measure of the precision required of the objective
% function at the solution. Both this and the previous condition must
% be satisfied for termination.
%
% OPTIONS(14) is the maximum number of iterations for the shadow
% targets algorithm; default 100.
%
% See also
% RBF, RBFERR, RBFFWD, RBFGRAD, RBFPK, RBFUNPAK, RBFSETBF
%

% Copyright (c) Ian T Nabney (1996-9)

% Check arguments for consistency
switch net.outfn
case 'linear'
    errstring = consist(net, 'rbf', x, t);
case 'neuroscale'
    errstring = consist(net, 'rbf', x);
otherwise
    error(['Unknown output function ', net.outfn]);
end
if ~isempty(errstring)
    error(errstring);
end

% Allow options to have two rows: if this is the case, then the second
row
% is passed to rbfsetbf
if size(options, 1) == 2
    setbfoptions = options(2, :);
    options = options(1, :);
else
    setbfoptions = options;
end

if(~options(14))
    options(14) = 100;
end
% Do we need to test for termination?
test = (options(2) | options(3));

% Set up the basis function parameters to model the input data density
net = rbfsetbf(net, setbfoptions, x);

% Compute the design (or activations) matrix
[y, act] = rbffwd(net, x);
ndata = size(x, 1);

switch net.outfn
case 'linear'
    % Sum of squares error function in regression model
    % Solve for the weights and biases using pseudo-inverse from
    activations
    temp = pinv([act ones(ndata, 1)]) * t;
    net.w2 = temp(1:net.nhidden, :);

```

```

net.b2 = temp(net.nhidden+1, :);

case 'neuroscale'
% Use the shadow targets training algorithm
if nargin < 4
    % If optional input distances not passed in, then use
    % Euclidean distance
    x_dist = sqrt(dist2(x, x));
else
    x_dist = t;
end
Phi = [act, ones(ndata, 1)];
% Compute the pseudo-inverse of Phi
PhiDag = pinv(Phi);
% Compute y_dist, distances between image points
y_dist = sqrt(dist2(y, y));

% Save old weights so that we can check the termination criterion
wold = netpak(net);
% Compute initial error (stress) value
errold = 0.5*(sum(sum((x_dist - y_dist).^2)));

% Initial value for eta
eta = 0.1;
k_up = 1.2;
k_down = 0.1;
success = 1; % Force initial gradient calculation

for j = 1:options(14)
    if success
        % Compute the negative error gradient with respect to network
        outputs
        D = (x_dist - y_dist)./(y_dist+(y_dist==0));
        temp = y';
        neg_gradient = 2.*sum(kron(D, ones(1, net.nout)) .* ...
            (repmat(y, 1, ndata) - repmat((temp(:))', ndata, 1)), 1);
        neg_gradient = (reshape(neg_gradient, net.nout, ndata))';
        end
        % Compute the shadow targets
        t = y - eta*neg_gradient;
        % Solve for the weights and biases
        temp = PhiDag * t;
        net.w2 = temp(1:net.nhidden, :);
        net.b2 = temp(net.nhidden+1, :);

        % Do housekeeping and test for convergence
        ynew = rbfwd(net, x);
        y_distnew = sqrt(dist2(ynew, ynew));
        err = 0.5*(sum(sum((x_dist-y_distnew).^2)));
        if err > errold
            success = 0;
            % Restore previous weights
            net = netunpak(net, wold);
            err = errold;
            eta = eta * k_down;
        else
            success = 1;
        end
    end
end

```

```

        eta = eta * k_up;
        errold = err;
        y = ynew;
        y_dist = y_distnew;
        if test & j > 1
w = netpak(net);
        if (max(abs(w - wold)) < options(2) & abs(err-errold) < options(3))
            options(8) = err;
            return;
        end
        end
        wold = netpak(net);
    end
    if options(1)
        fprintf(1, 'Cycle %4d Error %11.6f\n', j, err)
    end
    if nargout >= 3
        errlog(j) = err;
    end
end
options(8) = errold;
if (options(1) >= 0)
    disp('Warning: Maximum number of iterations has been exceeded');
end
otherwise
    error(['Unknown output function ', net.outfn]);
end
end

```

APPENDIX A14. NETLAB TOOLBOX [REF 24] CODE *rbffwd.m* USED TO SIMULATE OUTPUT OF TRAINED RBF NETWORK.

```
function [a, z, n2] = rbffwd(net, x)
%RBFFWD Forward propagation through RBF network with linear outputs.
%
% Description
% A = RBFFWD(NET, X) takes a network data structure NET and a matrix X
% of input vectors and forward propagates the inputs through the
% network to generate a matrix A of output vectors. Each row of X
% corresponds to one input vector and each row of A contains the
% corresponding output vector. The activation function that is used is
% determined by NET.ACTFN.
%
% [A, Z, N2] = RBFFWD(NET, X) also generates a matrix Z of the hidden
% unit activations where each row corresponds to one pattern. These
% hidden unit activations represent the design matrix for the RBF.
The
% matrix N2 is the squared distances between each basis function
centre
% and each pattern in which each row corresponds to a data point.
%
% See also
% RBF, RBFERR, RBFGRAD, RBFPK, RBFTRAIN, RBFUNPAK
%
% Copyright (c) Ian T Nabney (1996-9)

% Check arguments for consistency
errstring = consist(net, 'rbf', x);
if ~isempty(errstring);
    error(errstring);
end

[ndata, data_dim] = size(x);

% Calculate squared norm matrix, of dimension (ndata, ncentres)
n2 = dist2(x, net.c);

% Switch on activation function type
switch net.actfn

case 'gaussian' % Gaussian
    % Calculate width factors: net.wi contains squared widths
    wi2 = ones(ndata, 1) * (2 .* net.wi);

    % Now compute the activations
    z = exp(-(n2./wi2));

case 'tps' % Thin plate spline
    z = n2.*log(n2+(n2==0));

case 'r4logr' % r^4 log r
    z = n2.*n2.*log(n2+(n2==0));
```

```
        otherwise
            error('Unknown activation function in rbffwd')
    end

    a = z*net.w2 + ones(ndata, 1)*net.b2;
```

LIST OF REFERENCES

1. Lehr, Steven E, CDR USN, "Mine Warfare...An Enduring Challenge," brief given at Office of Naval Research Industrial Day, Jan. 1999.
2. Department of the Navy, ...*From the Sea: Preparing the Naval Service for the 21st Century*, Washington, D.C.: U.S. Government Printing Office, Sep. 1992.
3. Department of the Navy, *Forward...From the Sea*, Washington, D.C.: U.S. Government Printing Office, Sep. 1994.
4. U.S. Marine Corps Headquarters, *Operational Maneuver from the Sea*, Headquarters Marine Corps, Jan. 1996.
5. Department of the Navy, Operational Requirements Document, "Shallow Water Mine Countermeasures Mine/Obstacle Clearance Revision 2" (Draft).
6. The Humanitarian Organization People Against Landmines [http://www.mgm.org/index_e.htm], Dec. 2000.
7. Zamora, George, "Detecting Land Mines," [http://www.nmt.edu/mainpage/news/landmine.html], Sep. 1997, accessed Nov. 2000.
8. Dye, D, *High Frequency Sonar Components Of Normal And Hearing Impaired Dolphins*, Master's Thesis, Naval Postgraduate School, Monterey, CA, Sept. 2000.
9. Muir, T.G., Smith, D.E., Wilson, P.S., "Seismo-Acoustic Sonar for Buried Object Detection," Proceedings of the Symposium, *Technology and the Mine Problem*, Naval Postgraduate School, Monterey, CA, Sept. 1995.
10. Fitzpatrick, S.M., *Source Development for a Seismo-Acoustic Sonar*, Master's Thesis, Naval Postgraduate School, Monterey, CA, Dec. 1998.
11. Hall, P.W., *Detection and Target-Strength Measurements of Buried Objects Using a Seismo-Acoustic Sonar*, Master's Thesis, Naval Postgraduate School, Monterey, CA, Dec. 1998.
12. Sheetz, K.E. *Advancements in Buried Mine Detection Using Seismic Sonar*, Master's Thesis, Naval Postgraduate School, Monterey, CA, Dec. 2000.
13. Guy, J. *Seismic Sonar Developments for the Detection of Buried Landmines*, Research Project Report, Naval Postgraduate School, Dec. 2000.
14. Viktorov, I.A., *Rayleigh and Lamb Waves, Physical Theory and Application*, Plenum Press, 1967.
15. Ben-Menahem, A., and Singh, S.J., *Seismic Waves and Sources*, Springer-Verlag, 1981.
16. Foster, J., White, M., "Pattern Recognition Expert System for Mine Classification and Detection Sonar," *IEEE Proceedings on Energy and Information Technologies in the Southeast SOUTHEASTCON '89*, Vol.1, pp. 277-282, 1989.

17. Saragiotis, C., Hadjileontiadis, L., and Panas, S., "A Higher-Order Statistics-Based Phase Identification of Three-Component Seismograms in a Redundant Wavelet Transform Domain," *Proceedings of the IEEE Signal Processing Workshop on Higher Order Statistics*, pp. 396-399, 1990.
18. Hil, D., Frances, S., *Pattern Recognition and Prediction with Applications to Signal Characterization*, American Institute of Physics, 1996.
19. Brunzell, H., "Extraction of Discriminant Features from Impulse Radar Data for Classification of Buried Objects," *Geoscience and Remote Sensing*, Vol. 3, pp. 1285-1287, Aug. 1997.
20. Jain, A., Duin, R., Mao, J., "Statistical Pattern Recognition: A Review," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 22, No 1, Jan. 2000.
21. Zambartas, M., *Introduction to Hidden Markov Models and their Application to Classification Problems*, Master's Thesis, Naval Postgraduate School, Monterey, CA, Sep. 1999.
22. Leon-Garcia, A., *Probability and Random Processes for Electrical Engineering*, Addison-Wesley Inc., Reading, MA, May 1994.
23. Therrien, C., *Discrete Random Signals and Statistical Signal Processing*, Prentice Hall, Englewood Cliffs, NJ, 1992.
24. Nabney, I., NETLAB: Neural Network Software,
[<http://www.ncrg.aston.ac.uk/netlab/netlab>], accessed Aug. 2000, last modified Oct 2000.
25. Carevic, D., Chant, I., Caelli, T., "Feature Extraction and Classification of Minelike Targets from GPR Data Using Gaussian Mixture Models," *IEEE Proceedings on Information, Decision, and Control, IDC '99*, pp. 329-334, Feb. 1999.
26. Reynolds, D., *Speaker Identification and Verification using Gaussian Mixture Speaker Models*, *Speech Communication Magazine*, Vol. 17, pp. 91-108, Mar. 1995.
27. Moon, T., "The Expectation-Maximization Algorithm," *IEEE Signal Processing Magazine*, Vol. 13, Issue 6, pp. 47-60, Nov. 1996.

INITIAL DISTRIBUTION LIST

	No. Copies
1. Defense Technical Information Center 8725 John J. Kingman Road, Suite 0944 Ft. Belvoir, VA 22060-6218	2
2. Dudley Knox Library Naval Postgraduate School 411 Dyer Road Monterey, CA 93943-5101	2
3. Chairman, Code EC..... Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, CA 93943-5121	1
4. Engineering and Technology Curricular Office, Code 34 Naval Postgraduate School Monterey, CA 93943-5109	1
5. Prof. Monique Fargues, Code EC/Fa Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, CA 93943-5121	3
6. Prof. Thomas G. Muir, Code PH/Mt..... Department of Physics Naval Postgraduate School Monterey, CA 93943-5117	1
7. Prof. Roberto Cristi, Code EC/Cx..... Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, CA 93943-5121	1
8. Prof. Ralph Hippenstiel, Code EC/Hi Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, CA 93943-5121	1
9. LT Craig A. Wilgenbusch 1833 Fern Dr. San Diego, CA 92102	1

10. CPT Kraig E. Sheetz 1
RD#2 Box 608
Greensburg, PA 15601
11. Mr. Jeremie Guy 1
100 rue des Marmuzots
21000 Dijon
France